Selected topics in optical burst switched networks Performance analysis

Master Thesis by

Laura Capdevila Masdeu

École Polytechnique Fédérale de Lausanne (EPFL)

And

Universitat Politècnica de Catalunya (UPC)

EPFL Assistant :

Sébastien Rumley

EPFL Supervisor :

Dr. Christian Gaumier

UPC Supervisors:

Dr. Josep Solé Pareta,

Dr. Davide Careglio

and Oscar Pedrola Escribà

November 2010

Acknowledgements

To Sébastien

for his invaluable help and guidance along this project, his suggestions and support

To Christian

for having hosted me in the TCOM laboratory and make me feel at home during the whole stage

To Josep, Davide and Oscar

for having had confidence in me and for having motivated me to do this project

To my friends in Suisse

who have accompanied me in my adventure in Switzerland

and specially to Jessica, Juan Fran, Gil and the rest of «Mouton team » components, because they are "super-genial" and because with them I have discovered the best Switzerland landscapes.

To my friends in Spain

who have been supportive, attentive and patient all along, and because I know they are always there although I am far from them.

Finally, to my parents and my sister Judit

for their encouragement and their unconditional support and love.

Contents

Overview of the Report

This report is the result of the final year project carried out at the Laboratoire des telecommunications (TCOM), École Polytechnique Fédérale de Lausanne (EPFL) between March and October 2010 under the supervision of Dr. Christian Gaumier and the assistance of Sébastien Rumley.

The work presented in this thesis is based on the JAVOBS simulator, which was presented for Oscar Pedrola in [31]. This OBS network simulator has been built on top of the JAVANCO framework developed for Sébastien Rumley and other collaborators in TCOM laboratory.

The report starts presenting the basics of OBS paradigm in order to situate the lector in context and to present the operation possibilities of OBS networks and its problem of burst contention. It continues in chapter 2 with the explanation of network modeling and specifically for the particular model of JAVOBS. In the different sections in chapter 3, the selected topics are presented and treated in detail.
Each section of chapter 3 is divided in a presentation of the topic and the reason why it is interesting to work on it, the necessary implementation to introduce it in the Javobs tool, some simulation tests to validate and check the correct functionality of the component developed and some results and conclusions by simulation comparison.

Mainly, the selected topics are the following:

- The routing topic is treated with the purpose to define a proactive routing protocol based on load balanced-algorithm which operates as a contention resolution technique.
- The virtual topology design is treated in order to also alleviate and reduce the contention problem by the establishment of a particular set of dedicated lightpaths on the physical network.
- To provide new models of traffic for JAVOBS simulator, the traffic generation topic is raised. Traffic characteristics and self-similar properties are presented and a Markov-modulated traffic generator and a Self-Similar traffic generator are implemented and validated. Then, its impact in the network performance is evaluated and compared.

CHAPTER 1. OBS Basics

1.1 Introduction

Nowadays, there is an increasing demand of transmission bandwidth as a result of the data traffic growth. An interesting solution to consider is the **WDM** (Wavelength Division Multiplexed) that permits to simultaneously transmit data on multiple wavelengths on a single fiber with high throughput. But currently, its capability **is not completely exploited** in a whole network due to the slowness of the nodes to receive, process and send to the next node the data information.

For this reason, it's necessary to research in new optical network solutions. The principal aim is to obtain networks that work totally in optical domain: eliminating the OEO (optical-electrical-optical) conversion and working directly in optical domain.

There are three switching paradigms for WDM:
OCS (Optical Circuit Switching) [1] consists of setting up circuit connections (lightpaths) between the network nodes with optical WDM cross-connects (wavelength routers). Two-way reservation is needed to set up lightpaths.
The principal advantage is that optical buffer (OEO conversion) is no needed at intermediate nodes and the main constraint is the fixed and limited number of wavelengths (channels) per fiber and the impossibility to fraction them.

In **OPS (Optical Packet Switching)** [2] the data is assembled in optical packets and a header is created with control information. The data and its header are sent together in the same channel through the network. In each intermediate node (router), the control information is extracted and processed in the electrical domain whereas the data is buffered and switched in optical domain.
The best benefit is its efficient bandwidth utilization: the transmission resources (i.e. wavelength) are not dedicatedly reserved and they are shared by traffic from many sources. The inconvenient is the immaturity of this technology: nonexistent optical buffers, complexity in the hardware and high cost of the components.

OBS (Optical Burst Switching) [3,4] represents a balance between circuit and packet switching. It consists in the aggregation of multiple data packets into a burst. Then, this burst is transmitted without the need of any type of buffering at intermediate nodes. Thus, the OEO conversion is not necessary. In OBS, the control information is carried on a dedicated channel and separately from the user data channels. OBS is generally based on one-way reservation protocol.

1.2 Optical Burst Switching basics

In OBS, the user data is collected at the edge of the network, sorted based on a destination address, and grouped into variable sized bursts. Prior to transmit a burst, a control packet is created and immediately sent toward the destination in order to set up a bufferless optical path for its corresponding burst. After a delay called offset delay time, the data burst is transmitted.

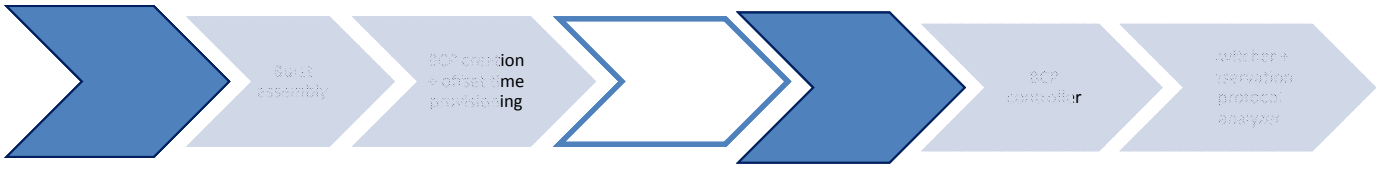
The network architecture is the framework that specifies the physical network components and their functional organization and configuration. It is composed by several nodes and links that connect these nodes, transporting data bursts on different wavelengths simultaneously (WDM).

In OBS networks, there are two different philosophies for defining the node architecture, which are the unification of the node functionalities in a single node element [5] and the differentiation of the node functionalities forming two types of nodes (edge and core nodes) [6,7 and many others].
The most common philosophy considered in the OBS literature is the second one. In this approach, the OBS network consists of interconnected core nodes that transport data incoming and exiting at the edge nodes.
The main characteristics and functionalities of edge and core nodes are:

- **Edge Node** (ingress node): The edge node is the element that introduces traffic in the network. Its main functions are: aggregation and assembly of data from client networks into data bursts according to the destination address with a separate queue for each possible destination; generation and emission of the burst control packet (BCP); offset time provision and burst transmission.

- **Core Node**: The principal function is to switch bursts. Core nodes have the responsibility to process the control packet (through its electronic conversion and its control information extraction), actualize and send it to the next node and configure the optical cross-connect (OXC) to switch the corresponding expected burst. If the reservation of the source is not possible, the data burst is dropped, which means that the burst cannot continue to travel across the network and its containing data cannot reach the destination and it is lost.

A functional block diagram of an OBS network transmission and based on nodes differentiation is presented in Fig.1.



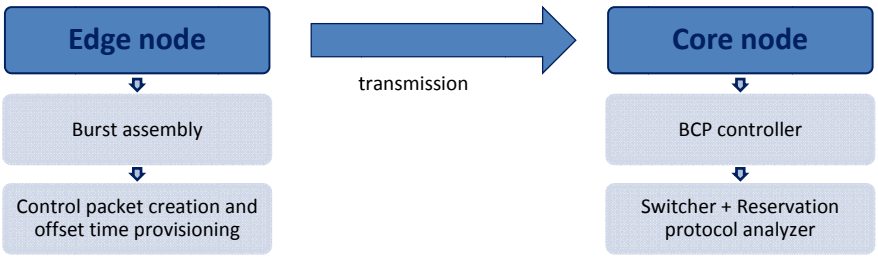


Figure 1 Functional block diagram of an OBS network transmission based on nodes differentiation

1.2.1 BURST ASSEMBLY

Burst assembly is the procedure of classification and aggregation of data packets from various sources (data from client networks) into optical bursts of variable length according to destination.

There are different assembly mechanisms [8]:

- Timer-based: after a fixed time, all packets present in the buffer are assembled into a burst.
- Burst-length-based: a maximum burst length is defined, so the burst is sent when buffer is filled.
- Hybrid timer/burst-length-based

The assembly algorithm is critical to the OBS network performance and its choice affect in the resulting OBS traffic statistic properties (burst length and interarrival time distribution).

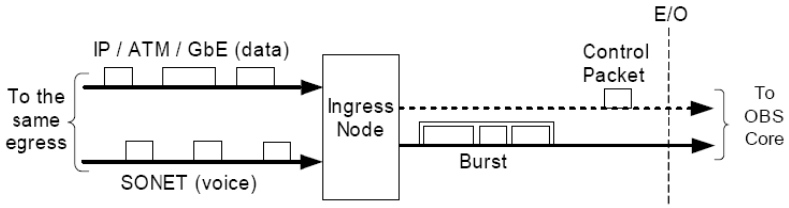


Figure 2 Burst assembly [3]

1.2.2 CONTROL PACKET GENERATION AND OFFSET TIME PROVISIONING

For each burst assembly unit a **control packet** is created and sent to the destination through the network with the objective to establish an optical path (wavelength reservation). After an offset time, the corresponding burst is sent on the same path without waiting any acknowledge (ACK) from the destination node (in the one-way signaling scheme case).

It's important to remark that the control packet is sent in a dedicated control channel, so the control and data are separated in space (using different channels), and in time (offset time between the transmission of

data burst and its control packet). One control channel can be associated and used for several data channels.

The control packet contains routing information, the burst length and the offset time. In each network node the control packet is converted to OEO and processed (electronically), while the data burst is switched all-optically.

Offset time

The offset time is the interval time between control packet and data burst emission. Its principal objective is to be consumed, so that control packets arrive at the destination just before that the data burst. The offset time value is actualized at each intermediate node in the BCP.

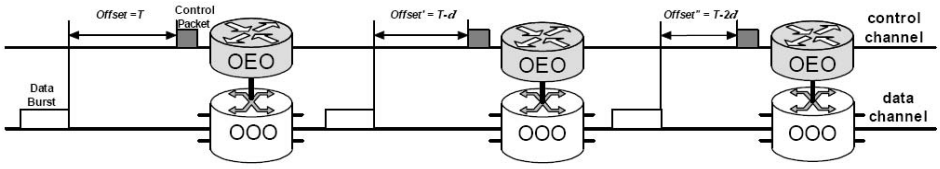


Figure 3 Control packet and offset time [3]

The offset value depends on the number of nodes that the burst has to traverse to the destination node (k) and potentially, on the channel reservation scheme, as we will see. The minimum offset time value for assuring that the control packet arrives before the data burst is:

$$T_{offset_{min}} = k \cdot T_{setup} + T_{OCX}$$

Where T_{OCX} is the time in the OXC for configuring its switch element in order to connect an input to a specific output (switch process time); and T_{setup} is the time to process the control packet, analyze its content and forward it (setup time).

In other words, the offset time is the time between the sending of the last bit of the control packet and the first bit of the data burst.

There are two different models to defining the offset time:

- **C-OBS** (Conventional OBS) [10]: the offset time is introduced in the edge node, so the data burst is waiting in the edge node that offset time passes. Offset time compensates the processing and switching time of all the intermediate nodes that exists in the routing path.
- **E-OBS** (Emulated OBS) [10,11]: the offset time is emulated in each core node using additional fiber delay line (FDL). Offset time compensates the processing and switching time only in the corresponding core node.

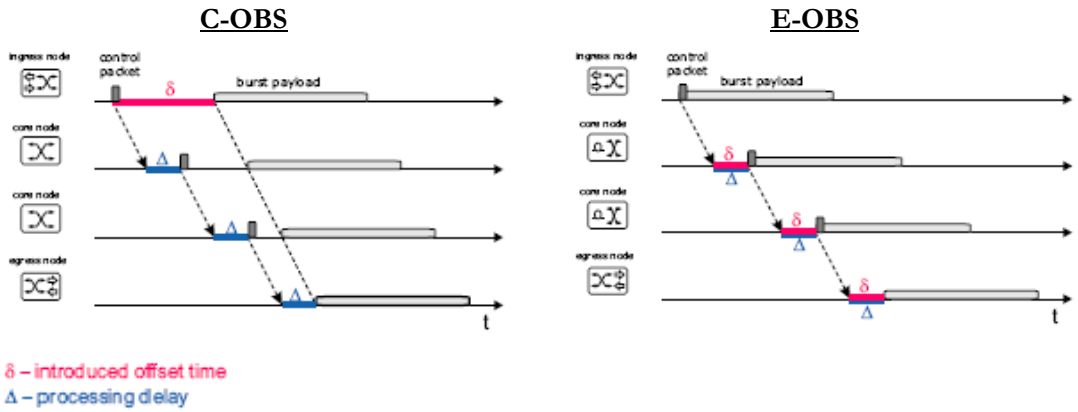


Figure 4 Models C-OBS and E-OBS [20]

1.2.3 SIGNALING PROTOCOLS

The **Signaling scheme** is the manner by which the nodes communicate connection requests, so it is used to set up a burst transmission path through the network and also tear down the connections.

OBS can use two different schemes for signaling a wavelength (bandwidth) reservation:

- **One-way signaling** (one-way reservation) scheme is the procedure to set up a burst transmission path through the network without waiting for a positive acknowledgment that assures the success of the reservation. The data is sent after the control packet without being sure that any reservation can be done. The one-way signaling scheme, is also known as **Tell-and-go (TAG) [12]**
  The reduced time for setting up the burst path permits to operate in large-distance networks, because the longer accumulate time for waiting the acknowledgment is not necessary, permitting to save time.

- **Two-way signaling** (two way reservation) scheme consists in end-to-end connection set up, so the edge node sends a control packet to make the reservation and waits for a positive acknowledgment to come back before transmitting data. The data is sent when the reservation is guaranteed.
  This scheme is also known as **Tell-and-wait (TAW) [13]** because the edge node sends the control packet to the destination node, waits the receiving of a positive acknowledge and send the data burst. For sending a positive acknowledge, it is necessary that all the intermediate core nodes traversed by the control packet perform the corresponding resource reservation.
  This scheme provides guaranteed transmission once the positive acknowledgment is received, but if the reservation cannot be achieved a long delay is introduced.

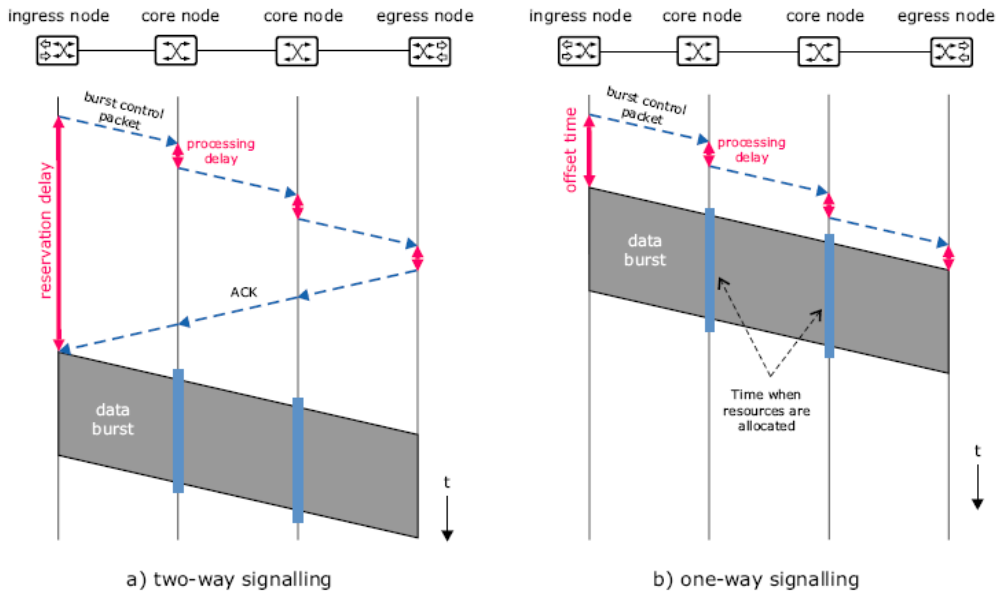


Figure 5 Signaling protocols: two-way and one-way signaling[24]

1.2.4 RESERVATION PROTOCOLS

The function of the **reservation protocols** is to prepare the switching resources (basically optical cross connects) and to make a booking of a wavelength (channel) in the output link for the incoming burst. If the wavelength is occupied by another burst, a contention resolution mechanism, if any exists, is applied.

A reservation is failed (and the burst is lost) if all the wavelengths in the link are reserved in the period of the burst transmission.

The different reservation protocols differ fundamentally in the way the wavelength is reserved or released[7]:

- **Explicit setup**: once the control packet is processed an immediate wavelength reservation is done, and an optical cross connect is configured.
- **Estimated setup:** the wavelength reservation and switches configuration are delayed until the arrival of the current burst in the core node.
- **Explicit release:** an explicit END control packet is send when the burst transmission is finished or there is some burst end detection mechanism.
- **Estimated release**: calculated using the burst length contained in the control packet. In this way nodes can know in advance when the burst finishes, and thus accept other reservations on the same channel prior to receive the corresponding burst.

It's important to know the burst assembly mechanism used. For example, in case of variable burst length (timer-based assembly mechanism), it is not possible to estimate the release time, except if the control packet contains length information.

Based on this classification and considering the previous point, several **one-way reservation protocols** have been proposed and compared in the literature to insure the allocation of resources and the optical switch configuration:

- **Just in time (JIT)** [15,16]: uses immediate reservation scheme. Just when the control packet arrives to the node, the output bandwidth for the incoming burst is reserved and at the end of the burst, it is released. If the reservation is not possible at this moment, the control packet is refused and the burst is dropped. This scheme follows the FCFS (First Come First Served) order.

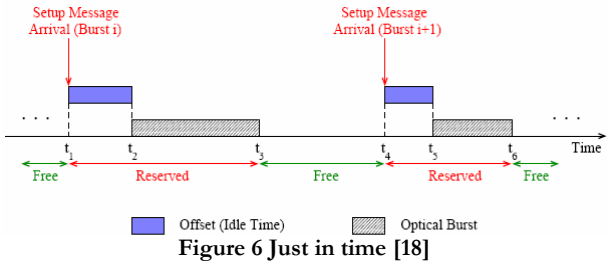


Figure 6 Just in time [18]

- **Just enough time (JET)** [3,17]: The wavelength is reserved during a period. This period starts when the data burst arrives to the node and finishes when the burst has been switched. It is the most popular architecture used. In this case, there is not bandwidth waste because the reservation is only for the duration of the data burst, but its disadvantage is the complexity of the reservation analysis when many channels are used.

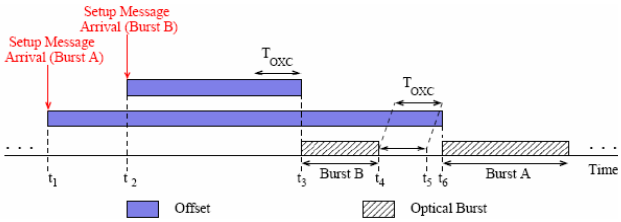


Figure 7 Just enough time [18]

- **Horizon** [4,19]: has a similarity with JET, so it uses a delayed reservation technique. The difference between them is that Horizon does not use the voids between the bursts and it makes the reservation of the burst according to a time horizon. The time horizon value is the earliest time after which there is no planned use of the wavelength (channel) and the output wavelength is reserved only if the arrival time of the burst is later than its time horizon.

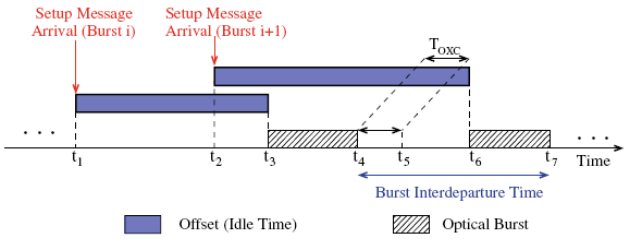


Figure 8 Horizon [18]

In [14,18], a detailed analysis of the JIT, JET, and Horizon wavelength reservation protocols is presented. In [18], a new reservation protocol called JIT+ is presented. JIT+ attempts to improve JIT by making a delayed burst reservation on a wavelength, although it is busy at the moment.

In the following table there are the main characteristics of the presented reservation protocols and its classification according to the wavelength reservation:

Reservation PROTOCOL	Reservation period	Way in which the wavelength in the outgoing link is reserved for bursts	Mechanism of reserve [17]	
			SETUP	RELEASE
Just in time (JIT)	Finishes when the burst has passed	Immediate reservation	Explicit setup	Explicit or estimated release
Just-enough-time (JET)	Transmission time of the burst	Delayed reservation (reserve a fixed duration)	Estimated setup	Estimated release
Horizon	Offset time + transmission time of the burst	Delayed reservation (reserve a limited duration)	Explicit setup	Estimated release

1.2.5 WAVELENGTH SCHEDULING ALGORITHMS

Wavelength channel scheduling algorithm uses the information extracted of the control packet to determine the wavelength channel which should be reserved. This process is realized in the core nodes, and it must take into account the existing reservations in each wavelength.

There are several algorithms proposed in the literature [4]. These algorithms can be classified into two groups according to the use of the possible voids between previous reservations caused by delayed reservation or a difference on offset time length.

- **Void-filling:** it utilizes and minimizes the voids between other reservations in the bandwidth if it is possible, so it tries to schedule a burst in one of the voids. The principal aim is to increase the bandwidth utilization, but requires more information and computation, so it's more complex.
- **Without void-filling:** It does not exploit the void intervals created by previous wavelength scheduled bursts.
- **Firs-fit Unscheduled channel:** It use the first unscheduled free channel.

In the following table there are some proposed algorithms for OBS networks:

Algorithm	Type	Reservation protocol based on	How it works?
LAUC [16] (Latest Available Unscheduled Channel)	Without void filling	Horizon	Analyses time horizon
LAUC-VF [19] (Latest Available Unused Channel With Void Filling)	Void-filling	JET	Analyses smallest gap
FFUC-VF [6] (First Fit Unscheduled channel)	First-fit	JIT	Analyses boolean (available or not)

1.2.6 NETWORK ROUTING

Routing consists in choosing a path for a burst from its source to its destination through the OBS network. According to the number of possible paths between two nodes, there are:

- **Single-path routing**, in which there is a single path between any pair of nodes in the network, so all the traffic between them must use this unique path.
- **Multi-path routing**, where several paths are possible between pairs of nodes. It is important to distinguish the following multi-path routing schemes:
    - **Alternative routing:** All the traffic might be sent over a primary routing path. If the primary path is unavailable, a secondary alternative path is selected. A particular interesting case of alternative routing is the **deflection routing**. It consists in locally deflect contending bursts towards links with available capacity. The routing decision is done in each core node.
    - **Multi-path source routing:** only the source makes the routing decisions. The information of the entire paths (possibly including deflection options) is computed by the source and it is included in the control packet.
    - **Load-balancing routing:** balance traffic over the network by using the multi-path scheme. It is possible to have load-balancing deflection routing or load-balancing source routing.

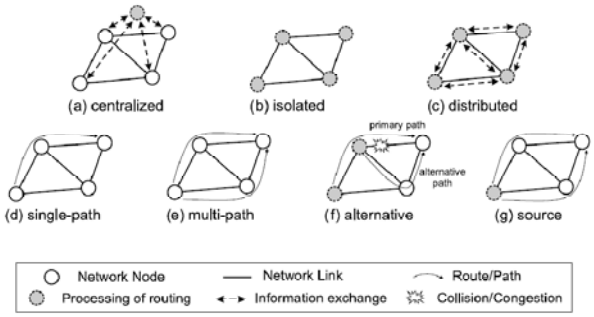


Figure 9 Routing algorithms [20]

The routing algorithms are classified as non-adaptive and adaptive algorithms.
In **non-adaptive** algorithms, also called *static*, the possible routes are calculated a priori (using static metrics: for instance physical distance or number of hops). Static routings are generally based on a shortest path routing algorithm.

The routing selection can be based in some fixed order (path rank) or based on a splitting scheme (i.e. the traffic is divided into fixed fractions and each traffic fraction is sent through each of the corresponding routing paths). Once the routing decision is taken, the route/s is/are fixed and does/do not change during the time.
The non-adaptive use is favorable when the traffic is stable and it does not change over the time, but when traffic fluctuates over the time, routes might become inadequate.

In **adaptive** algorithms, also called *dynamic*, the routes can be calculated a priori as in the static case or they can be re-calculated periodically according to traffic information (link congestion or number of burst contentions) in order to adapt to actual network conditions.
The path selection is dynamic and based on the exchange of congestion state information between the nodes, so the route is selected according to the changes in topology and the actual traffic.

In adaptive techniques, the routing decision can be done considering threshold traffic values or based on a list with all possible paths sorted for priority (path rank). Another possibility is to divide the traffic into fractions and sent them to routing paths according to their dynamically calculated congestion states (traffic splitting).

In [20], three adaptive families are distinguished. The adaptive routing is centralized (or global) when one centralized entity takes decisions according to the information collected from the entire network; is isolated (or local) if each node takes decisions using the local congestion information; and is distributive if uses global and local information.

In **OBS networks**, the routing algorithms are designed to minimize the network congestion, and consequently reduce the burst loss probability. For this reason, greater part of the research in OBS networks focuses on routing algorithms.

1.2.7 BURST CONTENTION IN OBS NETWORKS

Because of the limitations on the number of wavelengths that can be used in a fiber link, and hardware constraints at the network nodes, it is often not possible to set up a dedicated channel between every pair of nodes in the network. For this reason, the physical resources have to be shared and some problems of no availability appear.

Burst contention occurs in OBS nodes when two or more bursts attempt to go to the same output port[1] at the same time.
The simplest strategy is to process all bursts the same way. In case of contention, the latest burst that arrives is dropped.
To solve the burst contention problem, several strategies based on alternative forwarding in wavelength, time, and/or space domains have been proposed. The contention resolution strategies are classified as reactive techniques or proactive techniques:

The **reactive techniques** try to resolve burst contentions rather than avoid the contentions. Usually, they use local information of the node. These reactive techniques are:

- *Wavelength conversion (WC)* [21], in which contending burst can be sent on another wavelength through wavelength conversion. Its drawback is the immature and expensive technology. However, largely is assumed in the literature.

- *Fiber delay line Buffering (FDL)* [22] consists in delaying the burst a fixed time in a fiber delay line, but it supposes an extra delay and node complexity.

- *Deflection routing (DR)* [23,24,25 and others] consists on rerouting one individual burst over one single hop (hop-by-hop routing), so in case of burst contention in the primary port, the burst is deflected to an alternate not occupied port.

  The routing decision is taken at each core node (using local information) and only takes into account the selection of the next node. It is based on the information stored in routing tables (built with shortest path algorithm).

[1] A port is a pair of nodes in the network connected physically. It is directional, so if a fiber link connects nodes 0 and 1 there are two ports associated to it: port 0-1 and port 1-0

The selection of the alternative port could be: randomly, according to an arbitrary order or depending in a list of alternative ports assigned to each primary port.

To limit burst number hops, a TTL (time-to-live) field is included in the burst control packet.
In [23] deflection routing strategies are presented and compared considering synchronous and asynchronous burst interarrival times: DR (deflection routing), RR (Reflection routing), LBRR (load Balanced Reflection routing), RDR (Reflection-deflection routing) and MTR (multi-topology routing).

Some interesting observations of deflection routing are the following: it improves OBS behavior in networks with low traffic [26], but with high traffic it intensifies burst losses [29];
The out of order arrivals (large buffers are required in the receiving edge routers for reordering) and the burst transmission delays [27] are drawbacks to be considered;
Deflection routing transposes the congestion to another link rather than solve it [28]

The **proactive techniques** reduce the number of burst contentions and use feedback information that indicates the congestion state of the network.

- *Load-Balancing (LB) routing* [30] is a proactive technique which reroutes all the bursts over a whole path. If the path is saturated, traffic can be deflected over an alternate path. In consequence, it reduces the number of burst contentions by routing traffic over a less-congested part of the network.
  Load-balancing routing optimizes the routing decision according to the anticipated traffic demands (off-line) and the measurements of the congestion state of the network using feedback information.
  Its main objective is to minimize the burst loss probability by balancing the burst traffic load across the networks links.
  Two types of load-balancing routing algorithms are assumed: *non-adaptive*, which is based on static calculation and selection routes; and *adaptive* with its dynamic decisions.
  The advantage of load-balancing routing is that no additional hardware is required. Contrary, its main drawback is that it forces data to travel over one path to prevent congestion on another, which implies less flexibility if source routing is assumed.

Then, the effectiveness of the contention resolution scheme used depends on the traffic load offered to the network (traffic pattern) and the dimensioning of nodes and links (network topology). For example, reactive isolated alternative routing algorithms might be appropriate for small and medium networks whereas proactive routing algorithms are appropriate for large networks, because of the necessary collection of information about the network state to correctly distribute the traffic over the network.

Several reactive and proactive routing algorithms have been proposed for OBS networks in the literature. In [20] there is an OBS routing strategies classification with its literature references.

1.2.8 PERFORMANCE ANALYSIS

Quality of Service (**QoS**) is the service quality perceived by end users, so it measures how good the offered networking services are.
The primary QoS metric of interest in an OBS network is the burst loss rate (BLR), which represents the congestion state of the network.

Formally, the burst loss probability or burst loss rate corresponds to:

$$BLR = \frac{Number\ of\ lost\ bursts}{Number\ of\ emmited\ bursts}$$

Principally, in OBS networks the burst are lost because of the failure on resources reservation, which means that there is a greater number of simultaneous reservation attempts than number of available resources. Other possible causes of burst lost are the early arrival of data bursts by offset time and also due to simulation hazard.

It's important to consider that the dropped bursts have consumed and wasted network resources, so it affects to network throughput. For this reason, it is necessary to search the manner to diminish the possibility of not succeed in reserving resources at intermediate OBS nodes.

CHAPTER 2. Javobs simulator model

Currently, there are three different options in the study of OBS networks. The first one is to work on a prototype and use the statistical observation (empirically), but it requires money and time, and the prototype might be impossible to realize. The second option is to work on mathematic models (analytically), but implies an excessive complexity and difficulty to model the details. Finally, the third option and the most suitable approach is to work on a functional model through a simulation tool.

The simulators are based on abstract models which permit to conduct experiments with certain conditions. To run an experiment implies to simulate this experiment according to network protocols which are defined in the model. The output results of the simulation are then analyzed and interpreted with the purpose to know some aspects of network performance and capacity of various protocols, algorithms and network topologies.

In Fig.10, the basic simulation process for a OBS network experiment is represented.

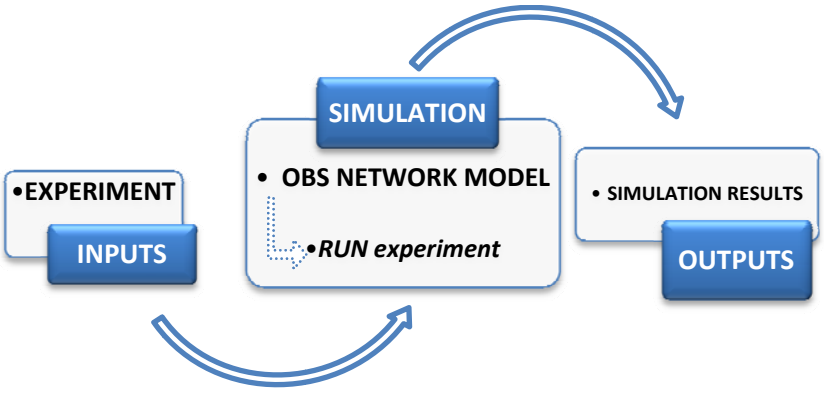


Figure 10 Basic simulation process

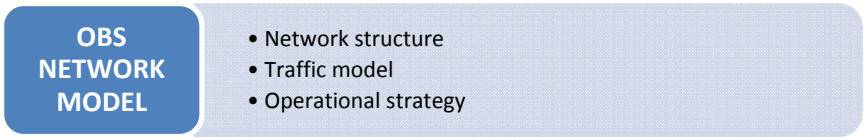
The work presented in this thesis is based on Javobs simulator. **JAVOBS** [31] is an OBS networks simulator that has been built on top of the **JAVANCO** framework. They are implemented in Java language and developed in the TCOM laboratory at EPFL with the objective to conceive powerful tools for network simulations.

In the following sections, the modeling of OBS networks, the experiment definition and the results analyzes are treated, and specifically for Javobs with the purpose to know its implementation and basic operation, establishing the basis for better understand chapter 3.

2.1 Modeling OBS networks – Javobs model presentation

An OBS network model is a simplified representation of OBS network. It contains the essential of a hypothetical real system to reproduce particular network behavior with the adequate accuracy.

The network **model definition** includes basically the definition of a network structure, a traffic model and an operational strategy.



2.1.1 NETWORK STRUCTURE

The network structure describe how data on a network is organized.

To define this data structure in OBS networks, it is necessary to consider the fact that a network is the physical interconnection of nodes using fiber links, that each fiber link supports a specific number of wavelengths (or channels) and that every pair of nodes is connected optically using certain of these channels.

Then, this information is organized in a physical and a optical levels:

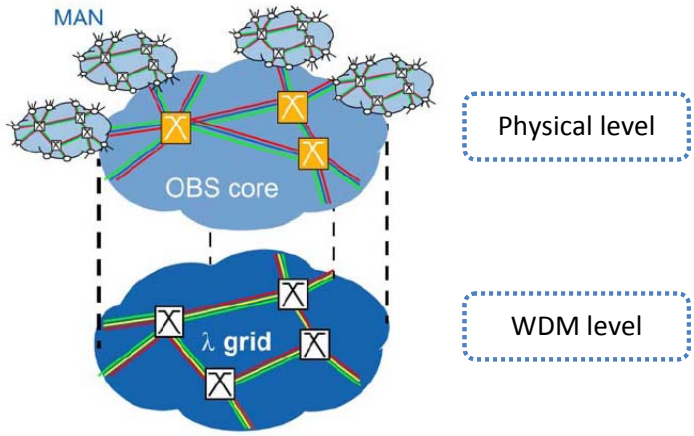
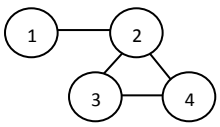


Figure 11 Network structure (physical and WDM layer) [41]

The physical level includes the physical topology of the network, with its nodes and fiber links characteristics.

Otherwise, the optical level (or WDM level) manages information about the lightpaths, which corresponds to the the optical path between any pair of nodes in the network. These ligthpaths define explicit routing paths, and altogether operate as a routing layer.

The ligthpaths are limited for the number of distinct wavelengths an optical fiber can carry. The information about the number of available channels per fiber link (capacity C) depends on the existing physical connections (topology) and is collected in the capacity matrix (Fig 12), which is interpreted for convention as showed in Fig (13).



$$A_{NxN} = \begin{pmatrix} 0 & C & 0 & 0 \\ C & 0 & C & C \\ 0 & C & 0 & C \\ 0 & C & C & 0 \end{pmatrix}$$

Figure 12 Example of capacity matrix

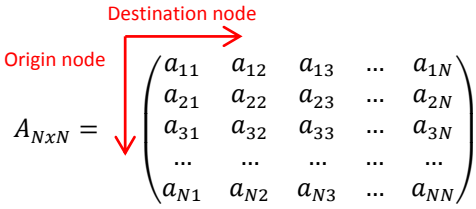


Figure 13 Interpretation of network matrix (convention)

For convention, the interpretation of a network matrix (Fig.13) is the following: N is the number of nodes in the network. Each element on the row represents the origin node index and each column represents the destination node index. Then, the element a_{ij}corresponds to the pair of nodes origin i and destination j, i.e. the connection i-j.

JAVOBS NETWORK STRUCTURE

In JAVANCO framework a central component for the management of network and topology information is defined as Graph Handler. This component uses different containers for collecting the information according to its type and function. There are defined layer, node and link containers.

Therefore, in Javanco the network structure is defined by a set of layers (physical and optical), and each layer is composed for a set of nodes and link containers with certain information. This information can be the nodes position, the pair of nodes a link connects physically, the optical routes or also other attributes which can be added for specific usage as link length or type of node, among other.

In JAVANCO the set of network information containers is defined in a xml file, but it can be designed and generated using graphical interface, a groovy script, a Java Object (topology generator) or directly writing the input xml file.

In JAVOBS the differentiation of nodes as a node architecture philosophy is assumed. For this reason, the network topology information associated to the Graph Handler (contained in the physical layer) is used to build the topology of the simulable network, which includes edge and core nodes.

In the simulable network the core nodes structure (in dark blue) corresponds to the physical layer topology and for each core node an edge node is connected as showed in Fig 14.

The edge and core nodes functionalities are the following:

Simulable element	Simulable node Functions:
Edge node	• Aggregator controller • Burst aggregator • Traffic generator • Offset time provisioning
Core node	• Switch controller • Reservation protocol • Signaling algorithm • Scheduling scheme

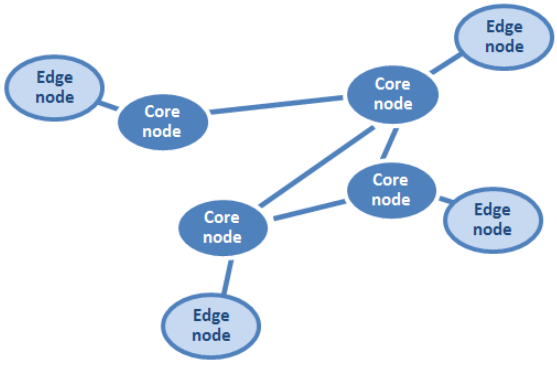


Figure 14 Javobs simulable network

In next section, the operation of the Javobs simulable network is exposed according to this functionalities.

2.1.2 TRAFFIC

In the traffic characterization, the principal aim is to obtain a traffic model that approximates the statistical behavior of real network traffic. So, it is related to burst arrivals in the network.

In general, **traffic generators** used in simulators proportion two sequences of random variables defining the burst interarrival times and the burst data sizes. In all the cases considered in this work, the burst data size is considered fixed and to define traffic only one time-serie of random variable is considered as a burst interarrival time.

JAVOBS TRAFFIC MODEL

The network traffic model is determined by the way the traffic is distributed over the network and the process type of the traffic.

The component which provides the traffic in the network is the **traffic provider** and it uses a traffic generator based on a certain traffic process. The implemented traffic provider provides Poisson traffic uniformly over the network.

In the most of studies the network arrivals are modeled as Poisson processes. In the real current networks, different traffic streams corresponding to different telecommunications services (voice, video, file transfer, etc.) are superposed (multiplexed) to form a heterogeneous mixture of traffic. This traffic mix fluctuates over all time scales. To represent it, a new concept of self-similarity has been considered in the traffic model conception and researchers are studying how to characterize it and how to decrease its bad affect in network performance.

In chapter 3, the traffic topic and this concept of self-similarity is treated and two other traffic models are implemented for Javobs simulator.

2.1.3 OPERATION STRATEGY

In terms of operation, it is possible to distinguish continuous models and discrete event models depending on how the states in the model are updated throughout the simulation.
In continuous simulation models, the state changes are made continuously as simulation time progresses. On the contrary, in a discrete-event model, changes only are done at discrete time points in simulation.

Most network simulators (as Javobs) use discrete event simulation, so an event list sorts pending events according to a chronological order and it controls the execution sequence of the event routines.

In Fig 15, the discrete-event model is represented:

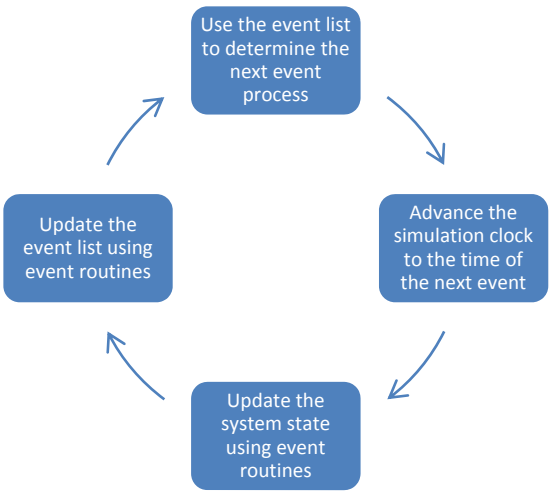


Figure 15 Discrete-event model

The idea is that repeatedly, the event list is checked for pending events.

As the event list is ordered, the time progress is emulated each time the clock is set to next event time.

Then, the corresponding event routines are executed and the system variables and the events list are updated. Finally a statistical report is generated providing the simulation results.

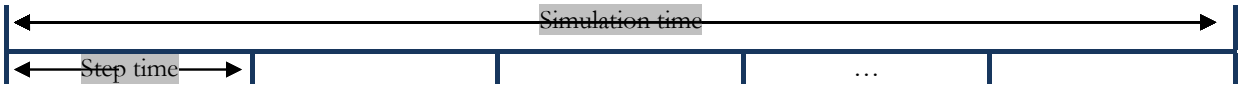
In simulation, the need of being able to reproduce an experience and recreate certain behavior is very important. Therefore, it is necessary to be able to reproduce a given random stream. To accomplish it, a **pseudo-random number generator (PRNG)** as random source (which takes a seed number as a parameter) is used. From the same seed, the same stream of random values is obtained, and consequently the same sequence of bursts is generated (for example).

In order to obtain higher fidelity to the results of our simulations, it is necessary to run each experiment several times changing the PRNG's seed. The purpose is to obtain several samples of the simulation results and average them, increasing the probability to obtain a results sample as near of the mean as possible. The results represented in this work, will be always the mean value of the results obtained for at least 5 different seeds.

JAVOBS OPERATIONAL STRATEGY

JAVOBS implements an OBS **discrete event simulation model**, so the network operation is represented as a chronological sequence of events and each event marks a change of state in the network.

The simulation is divided in steps for continuous time



In each step, and for each node the following actions are realized:

Edge node

- Traffic is generated: a set of bursts with destination assigned are obtained from the traffic provider

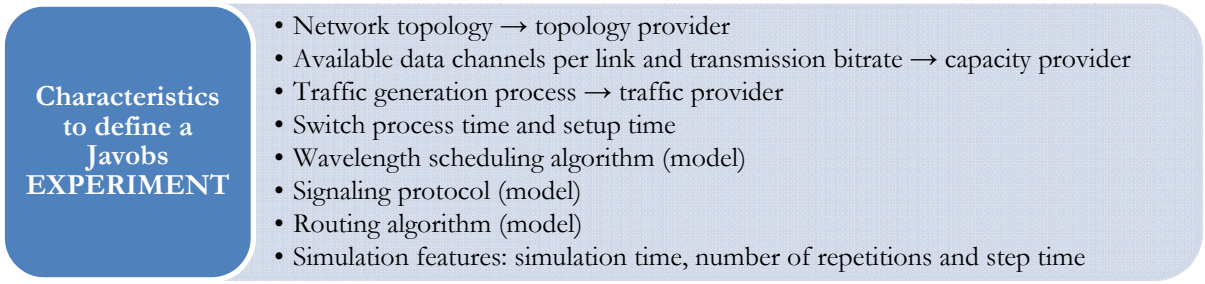
Core node

- Reservation list of all the transmission channels is updated and passed reservations are deleted.
- It is checked if there are some events that have to be executed in the current step, and they are executed in order. The execution of these events depends on the network protocols considered in the model:
  - Wavelength scheduling algorithm
  - Signaling protocol
  - Routing protocol

Finally, the obtained information during the step time is transferred to a data collector and the step time is decreased to the events that are waiting in the list. The wavelength scheduling algorithm and routing protocol are considered as part of the operational strategy of the model, but Javobs permits to configure and select them for each experiment.

2.2 Javobs network experiment definition

Actually, to simulate a JAVOBS network it is necessary to define an experiment by determining minimum the following characteristics:



Changing and playing with these options in different experiments, the simulator obtains different results for the same network permitting to evaluate and compare the performance and to determine weak points.

In Javobs, to define the physical characteristics of the network needed to build the corresponding simulable network (with core and edge nodes) it is necessary to use the **topology provider.** It is in charge to extract the network topology information contained in the xml file and to provide it.

As optical fibers can carry several channels simultaneously, each of them on a particular wavelength it is necessary to define the number of simultaneous wavelengths (channels) in each fiber link and its transmission bitrate. The **capacity provider** is the Javobs component which provides the network capacities (available data channels per link) and it fixes the nominal channel bitrate, which is the maximum traffic rate in a channel. It needs a capacity generator, which can be uniform or non uniform depending if all the fibers in the network have the same number of channels or not. The capacity provider is treated in section 3.2.

As mentioned, the **traffic provider** provides information of the traffic demands distribution in the network. It is in charge to generate the traffic (with a traffic generator) and distribute it over the network. The traffic provider is treated in section 3.3.

Currently, the possible wavelength scheduling algorithms in Javobs are LAUC, LAUC-VF, FFUC and FFUC-VF, which as evoked in chapter 1, they are based on Horizon, JET, JIT and E-JIT, JIT+ reservation protocols.

Javobs uses **one-way signaling protocol,** which means, as evoked on chapter 1, that once the control packet is created and transmitted to the destination node and after an offset time, the data burst is sent without waiting any positive acknowledgment for the network.

The actual routing scheme is called *simple routing logic* and it is based on shortest path algorithm, which searches the path with less number of hops considering all the possible path between each route in the network.

Besides the explicit experiment definition it is necessary to specify the characteristics of the simulation. As it has been said, Javobs is a discrete time simulator, so, it is also necessary to introduce the simulation time, the number of repetitions and the step time. It is possible to simulate in one time (Fig16) or to do the simulation several times with different time length. The second possibility is used and justified in traffic section, for the traffic experiment simulations in order to obtain more information of the obtained results overlapping them in time (Fig17 and Fig18).

In both cases, it is necessary to define the duration of each simulation and the step-time.

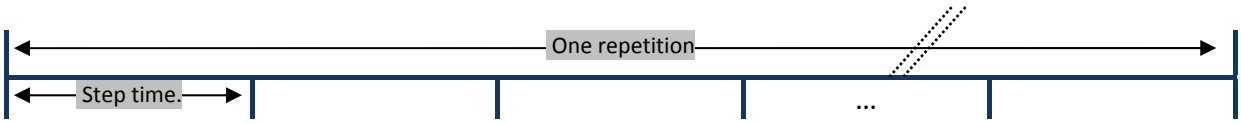


Figure 16 Simulation with only one repetition

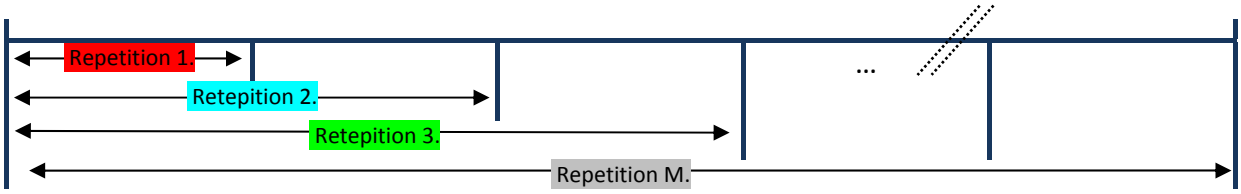


Figure 17 Simulation with M repetitions

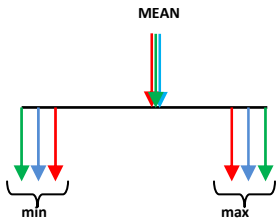


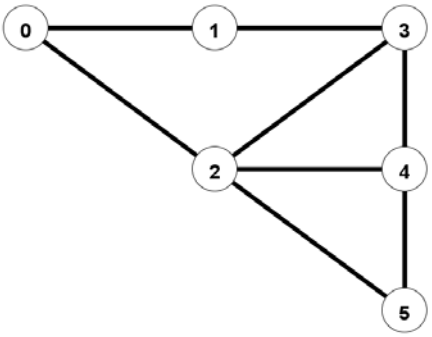
Figure 18 Simulation with M repetitions permits to obtain more information of the results (variability)

General assumptions for the simulations realized in this work:

In this work, all simulations are run under the following general assumptions:

- 16 bidirectional wavelengths with a transmission bitrate of 10Gbps per link;
- For simplicity, the switching and processing time are neglected.
- FFVF_Algorithm is used as a wavelength scheduling algorithm
- For all cases we consider fixed burst length size fixed to 1.2Mbits.
- The simulation results are obtained from 1 repetition of 400000 mseg with a discrete time simulator that works with steps of 20mseg of duration. Except in the traffic section, where the repetitions are overlapped.
- 5 different seeds are considered for the PRNStream.
- The burst data size is considered fixed and to define traffic only one time-serie of random variable is considered as a burst interarrival time. This fact is consequence to consider a timer-based assembly algorithm in Javobs simulator.
- Poisson traffic uniformly distributed, except in the traffic section.

In the simulations done, the **SIMPLE NETWORK topology** is used.



It has 6 nodes, 8 links connected as it is showed in the figure. In this thesis, for this topology and for the assumptions considered above, the network is considered:

- Low-charged → for $\varrho < 0.625$
- High-charged when the most of the link are saturated → for $\varrho > 1$

Its network load computation is included in Apendix 1.

2.3. Simulation results - Javobs results analysis

Once the simulation is done, it is necessary to analyze and interpret the obtained results. For achieving this, Javanco has different analyzers for filtering and extracting specific parameters from all the obtained data, and then it represents them in a graph using its graphical user interface.

These analyzers permit to observe specific parameters of the simulation results simplifying the selection and interpretation task. These analyzers are:

- The **DefaultPerformanceAnalyser**, which gives global information of the entire network: global BLR, number of bursts in the network, carried load (gbit) and offered load (gbit).

- The **PerPortAnalyser** which provides per port information: offered load, losses per port due to no unavailability and carried traffic.

- The **PerRouteAnalyser,** which provides per route information: local BLR, carried traffic and offered traffic per route.

For this work, a new analyser called **LinkTrafficDetailer** has been implemented with the purpose to give specific information of the traffic that circulates between each pair of nodes in the network. It is used to probe and evaluate the schemes implemented and it also facilitates the result analysis.

Given a specific port, the link traffic detailer provides information of the number of bursts sent, the number of burst well received and lost, the BLR per port and also the specific BLR per port and pair origin-destination.

As example, the fixed routing represented in Fig.19, which corresponds to the basic routing logic (based on shortest path), is used to prove the correct operation of the link traffic detailer.

In fig 20, the number of bursts per port detected by the link traffic detailer during the simulation are represented.

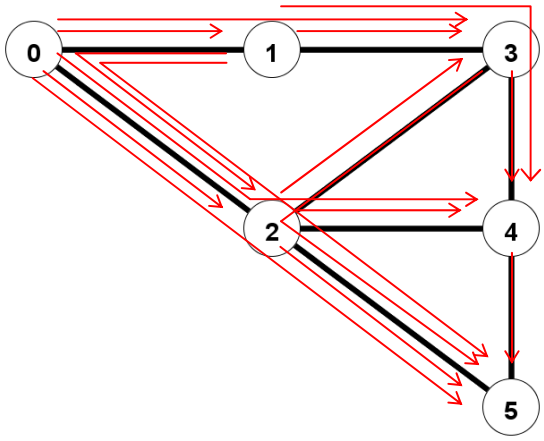


Figure 19 Basic routing logic on SIMPLE network

For this validation, only one port sense is considered because the results are symmetric. So, for the fiber link 0↔1 (bidirectional), the port 0-1 is treated but not the 1-0 port.

Observing the flows between a port in Fig.20 and the connections associated with this port in Fig.19, it is possible to corroborate that the link traffic detailer works as it was expected to.

For example, looking port 1-3, the flows which travel through it are 1-3, 0-2 and 1-4; which correspond to the obtained connections for the traffic detailer in this port. The same occurs for the rest of the ports.

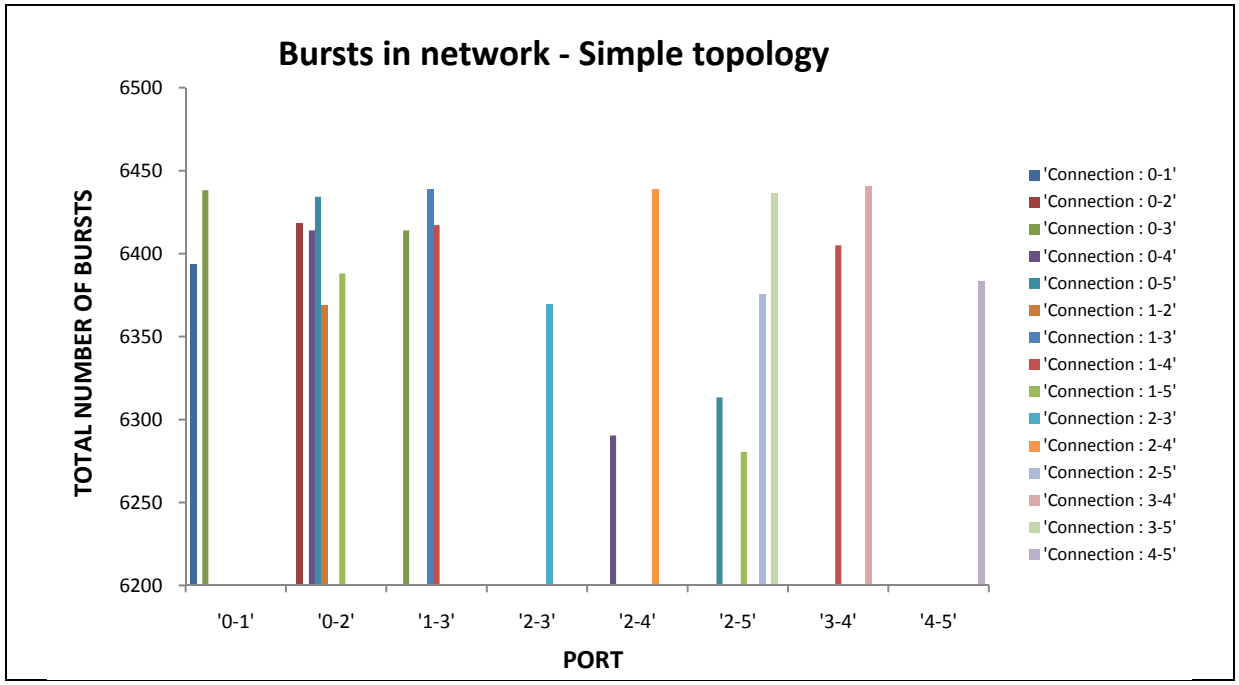


Figure 20 Bursts in network for simple topology and basic routing logic

CHAPTER 3. Selected topics in JAVOBS simulator

In this thesis, the three selected topics treated for JAVOBS simulator are virtual topology design, routing protocols, and traffic generators.

As it has been presented in chapter 1, one of the principal problems in OBS networks is the burst contention and, consequently its resulting higher burst lost probability that directly impacts on the service quality perceived by the end users.

The routing protocols are treated in section 3.1 with the purpose to evaluate them as a contention resolution technique. Today, JAVOBS offers the static shortest path routing scheme based on Dijkstra's algorithm, and it also offers the possibility to use a deflection routing algorithm. As the deflection routing algorithm is a reactive burst contention technique, it is interesting to design a new routing algorithm based on proactive routing and examine if the resultant network performance could be better. The proactive routing protocol example implemented is based on the load-balanced algorithm using multi-path source-routing characteristic, i.e. only the source makes the routing decisions and the fixed routes are included in the control packet.

Another method to alleviate and reduce the contention problem is to make use of dedicated wavelength for some determinate routes in the network. To evaluate this option, the virtual topology design is treated in section 3.2, and two virtual topology schemes for JAVOBS are proposed and validated.

In section 3.3, the traffic generation topic is raised with the objective to provide new generators of traffic for JAVOBS simulator. Actually, Javobs uses a traffic generator based on a Poisson model. However the standard Poisson traffic model does not model the real internet traffic very well. It is thus necessary to implement more sophisticated traffic generators. The Markov modulation is an interesting first step towards self-similar traffic models and for this reason it is studied in this work. Also, a traffic generator based on a more realistic model for self-similar traffic is treated.

3.1. ROUTING

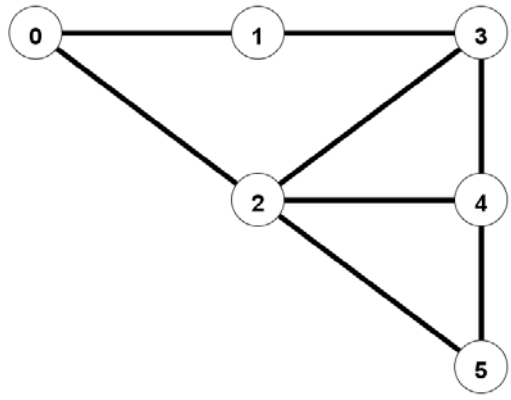
Analyzing the routing network mechanisms, the shortest path algorithm minimizes delay and optimizes the utilization of resources, but, does not take into consideration the traffic load offered to the network. For this reason, it sometimes causes burst contention in some links and underutilization in others.

In this work, a **load-balancing source routing scheme** is proposed. As all the load-balancing algorithms, it is based on the use of the network multi-path routing capability with the purpose to distribute the offered traffic over several paths, thus permitting to reduce the load on the more congested links.

The source routing characteristic gives to the source the control of the path/paths on which the bursts are forced to travel.

The principal idea is that the traffic is split over the network by sending a fraction of traffic over each candidate path between origin and destination.

As example, the SIMPLE topology and the routing information defined in the table (Fig.21) are considered. Assuming the route 1 → 5, there are three candidate paths to send the bursts through. For each emitted burst and depending on a random value, the source decides the explicit route that this concrete burst will take on its entire trajectory to the destination. In the example, the bursts emitted by node 1 can be sent to 3 with a probability of 0.6 (because two path of ratios 0.3 goes to 3) and to 0 with a probability of 0.3.



		RATIO	ROUTE
0 ↔ 1		1	0 → 1
0 ↔ 2		1	0 → 2
0 ↔ 3		0.5 0.5	0 → 1 → 3 0 → 2 → 3
0 ↔ 4		1	0 → 2 → 4
0 ↔ 5		1	0 → 2 → 5
1 ↔ 2		0.45 0.55	1 → 0 → 2 1 → 3 → 2
1 ↔ 3		1	0 → 3
1 ↔ 4		1	1 → 3 → 4
1 ↔ 5		0.333 0.333 0.334	1 → 3 → 4→ 5 1 → 0 → 2→ 5 1 → 3 → 2→ 5
2 ↔ 3		1	2 → 3
2 ↔ 4		1	2 → 4
2 ↔ 5		1	2 → 5
3 ↔ 4		1	3 → 4
3 ↔ 5		1	3 → 4 → 5
4 ↔ 5		1	4 → 5

Figure 21 Routing information for load-balancing source routing logic

Therefore, the routing information has to include the list of the possible paths between each pair of nodes in the network with its corresponding ratio. The ratio is a value Є (0,1) and corresponds to the probability to choose the route that is associated with.

It is necessary to take into account that once the route of a burst is decided for the source, it cannot be changed and has to be incorporated in the BCP corresponding to this burst.

3.1.1 LOAD-BALANCING SOURCE ROUTING - JAVOBS IMPLEMENTATION

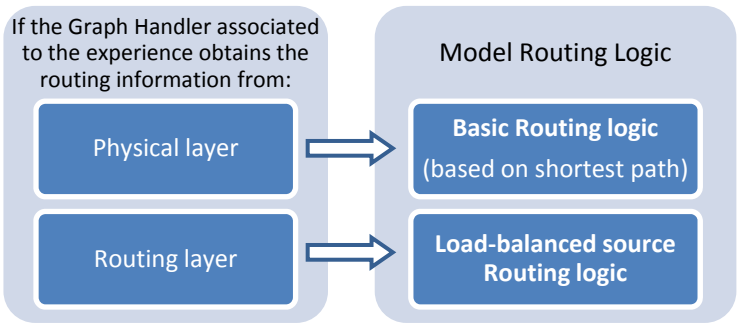
First of all, to put on Javobs the new routing information presented, it is necessary to associate a new routing layer to the network manager (Graph Handler). This routing layer has to contain information of the explicit route paths possibilities between nodes and the corresponding different ratios assigned to each route.

In this work, the xml file which defines the network has been manually modified to include the routing layer information as showed in Fig.22 (SIMPLE_VALIDATION_routing xml). In futures works, an algorithm for automatically calculate the best ratio to each path could be implemented given a determined topology network.

```
<layer id="routing">
        <!-- For describing routes: route [ratio, path] -->
        <link orig="0" dest="1" >
                <route ratio="1" path="0,1"/>
        </link>
        <link orig="0" dest="2" >
                <route ratio="1" path="0,2"/>
        </link>
        <link orig="0" dest="3" >
                <route ratio="0.5" path = "0,1,3"/>
                <route ratio="0.5" path = "0,2,3"/>
        </link>
                        ...
</layer>
```

Figure 22 Example xml to describe explicit routes and ratios in routing layer

Then, as it is implemented now, depending on the layer that the Graph Handler uses as a routing layer the built model will have a determinate routing logic:



Any of the routing logics used in Javobs has two principal functions:

- initialize the logic parameters (switching and processing time) and obtain the routing information from the layers associated to the agh → in Javobs implementation it corresponds to the "init" method of the routing logic class.
- obtain the next node to go given the current node and considering a determinate destination → it corresponds to "get next hop" method in the routing logic class.

To understand the Javobs implementation of load-balancing routing logic is necessary to first explain the Basic routing logic implementation.

The **Javobs Basic Routing Logic** (by the *init method)* initialize the parameters (switching and processing time) and collects the network information extracted from the "physical layer" of the Graph Handler associated to the experience and defined in the xml file. Then, all the possible paths between each pair of nodes in the network, their number of hops and shortest path are computed in order to establish and fix the next node to go given the current one. The computed information is included in a matrix, where each one of its elements corresponds to the next hop to do (interpreting the matrix as is presented in Fig.13).
When the method *getNextHop* is called, in the Basic Routing logic case, the next node to go is directly the corresponding element of this computed matrix.

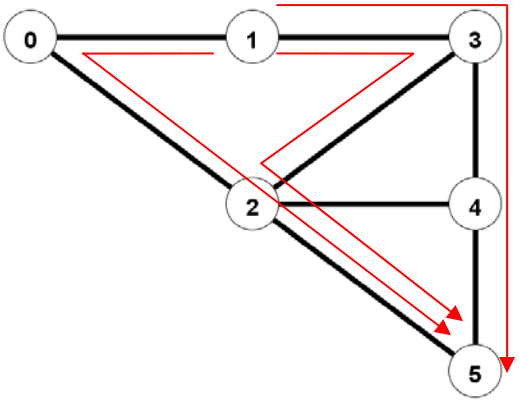
For the **Javobs Load-Balancing Routing Logic** case, the *init method* also serves to initialize the parameters and collect the network information, but in this case, the information is extracted for the "routing layer" instead of the "physical layer". This information corresponds to all the possible paths between each pair of nodes in the network and its corresponding ratio value and it is included in a general list of paths and ratios called *pathSetTotal* in this work.

In Load-balancing routing, the *getNextHop* method depends on the source routing decision. First of all the source node choose the path for which the burst will travel and fix it in the burst control packet (SourceRouteBCP), and once the path is fixed, *getNextHop* returns the next hop taking into account the fixed path included in the BCP.

For permitting this, it has been necessary to implement a new Burst Control Packet (*SourceRouteBCP*) for the load-balancing source routing logic, because the existent BCP component (*ObsBCP*) is not able to transport information about the fixed path from source to destination. Also a *SourceRoutedBCP Provider* is implemented to create a new *SourceRoutedBCP* instead a *ObsBCP* when a new BCP is required.

Continuing with the getNextHop method explication, given a pair origin-destination and considering all the possible paths between them, the idea is that if the current node corresponds to the origin node of the route, one path among all possible path between origin and destination is selected and is established in the SourceRouteBCP as a fixed route. The selection of the path is done using a random value obtained from the PNRGStream in the current step and the ratios associated to each of the possible paths.
If the current node corresponds to one of the intermediate nodes in a route already fixed, it returns the next nodes following the fixed path order.

In Fig 23, an example of the explained load-balancing routing operation is showed:



Supposing the connection 1→5, the route possibilities to consider are:

- 1 → 3 → 4→ 5 with ratio 0.33
- 1 → 0 → 2→ 5 with ratio 0.33
- 1 → 3 → 2→ 5 with ratio 0.34

} Information extracted from routing layer in xml file SIMPLE

When getNextHop method is called for first time, and the current node corresponds to the source node, if the random value is for example 0.4, the route fixed in the BCP is 1 → 0 → 2→ 5.

In next times, when the method getNextHop is called, the next node in the fixed path of the BCP is returned. Then, if the current node is 0 the next hop is 2, and if it is 2 the next hop is 5.
Once the route of a burst is fixed by the source, the next hop is extracted from the BCP.

Figure 23 Example of the load-balancing source routing operation

3.1.2 LOAD-BALANCING SOURCE ROUTING - VALIDATION

Using the link traffic detailer is easy to prove that the bursts corresponding to a determinate connection (origin, destination) travels through the expected ports and that they follow the route depending on its probability ratio.

For example, in the connection 1-2, the possible routes are 1→0→2 and 1→3→2 with ratios 0.45 and 0.55 respectively; it means that the bursts from 1 to 2 has less probability (45%) to go through the first path option than through the second one (55%).
As it is possible to corroborate looking the results obtained by the burst analyzer in fig24, the number of bursts in ports {1-0, 0-2} is 1950 and in ports {1-3, 3-2} is 2300 of the total bursts of 4250, which effectively corresponds to the 45% and 55%.

For the connection 1-5, in which the routes have the same probability to be chosen, it is possible to observe that the number of bursts in all the ports is approximately the same. In ports 1-3 and 2-5 there is the double of bursts because two of the three possible routes uses the ports 1-3 and 2-5 (as seen in fig 24).

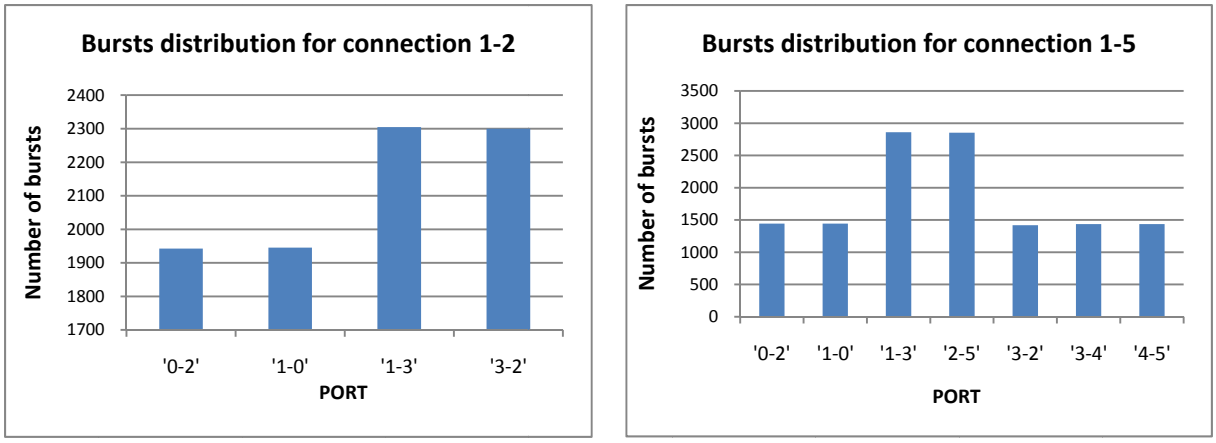


Figure 24 Number of burst obtained by burst analyzer for connections 1-2 and 1-5 using a load-balancing source routing

3.1.3 LOAD-BALANCING SOURCE ROUTING - SIMULATION AND RESULTS

The study consists of a simulation over the SIMPLE topology with the different routing logics in order to evaluate the new load-balancing source routing scheme and compare the performance obtained with the other routing logics in Javobs.

Presently, the routing logic options in Javobs are the presented in Fig.25 (the new routing option is differentiated with dark blue).

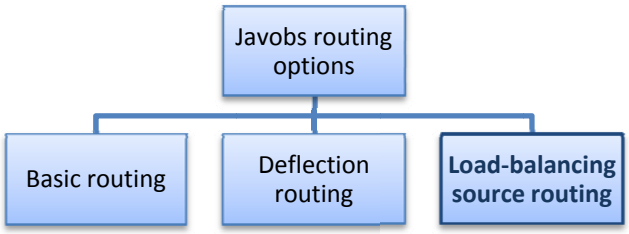


Figure 25 Routing logic options in Javobs simulator

As mentioned, the number of burst contentions can be reduced using deflection routing as a reactive technique or load-balancing routing as a proactive technique. Therefore, one interesting aspect to look at is the number of losses per port due to no unavailability of resources (Fig.26):

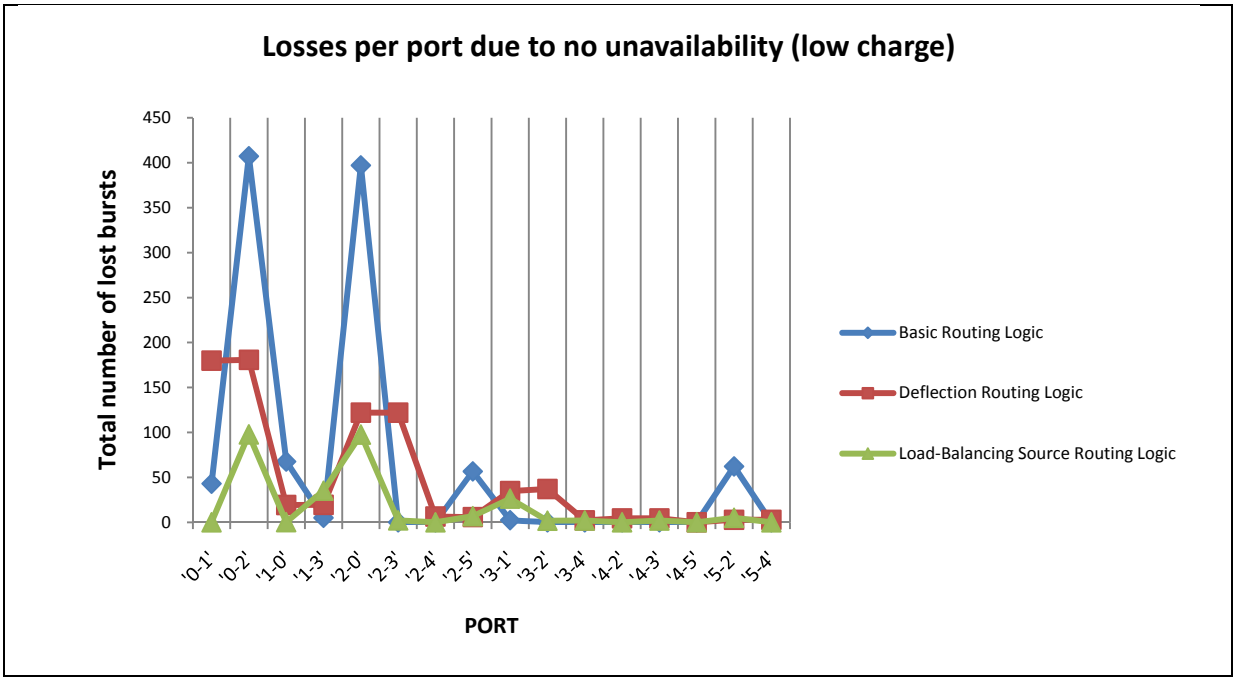


Figure 26 Losses per port due to no unavailability (charge=0.6)

As it is observable, the basic routing has problems of burst contention in some links (principally 0-2, 2-5 and 0-1) and underutilization in others. The deflection routing and load-balancing source routing schemes can reduce the number of lost burst due to the contention problem.

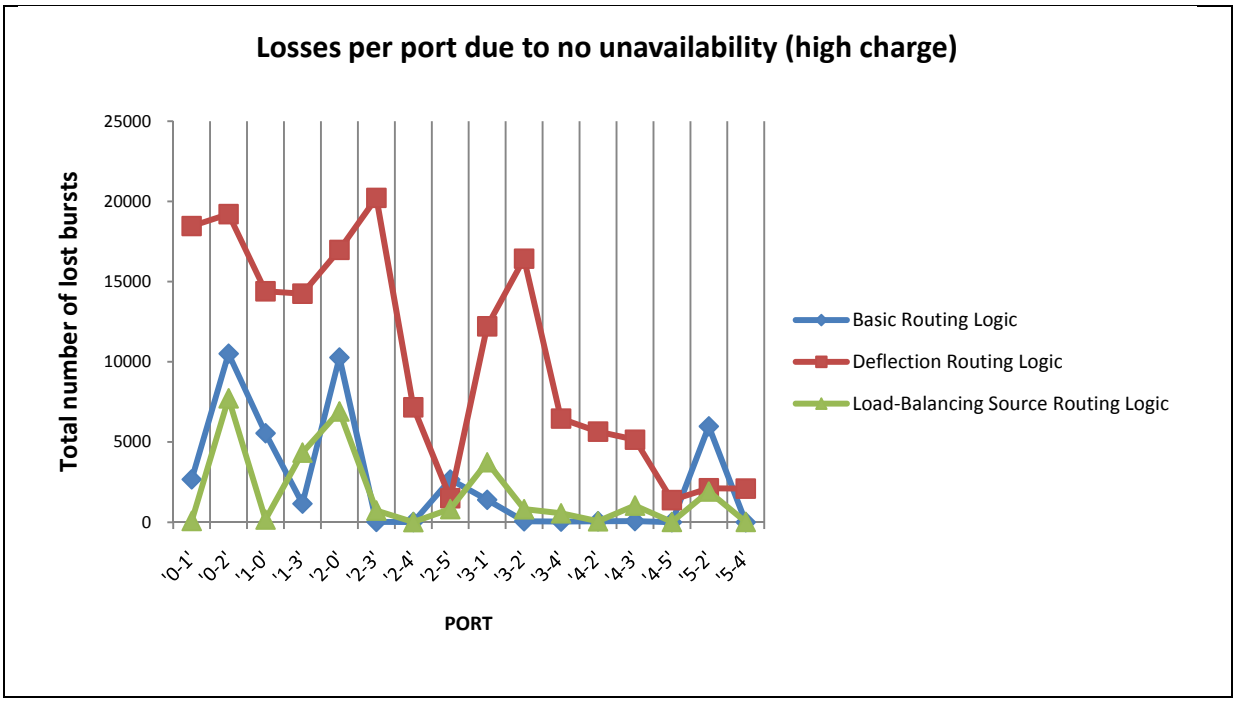


Figure 27Losses per port due to no unavailability (charge=1.2)

As the network charges increases the deflection routing logic scheme becomes worst in terms of number of lost bursts due to the no unavailability of resources, which, as commented in chapter 1, it is the

deflection routing drawback. For high network loads, the bursts deflect and deflect over the network entering a loop and they cannot reach the destination node. Otherwise, the load-balanced routing logic continues being better than basic routing logic in most of the ports (although the network load increases).

As BLR adequately represents the congestion state, it is the primary metric of interest in the routing study. The global BLR gives information of the entire network congestion and the BLR per route gives specific information of the congestion routes of specific routes.

In Fig.28, the global BLR obtained for each one of the routing logics (using the Default Performance Analyser) is plotted in function of the traffic load in the network.

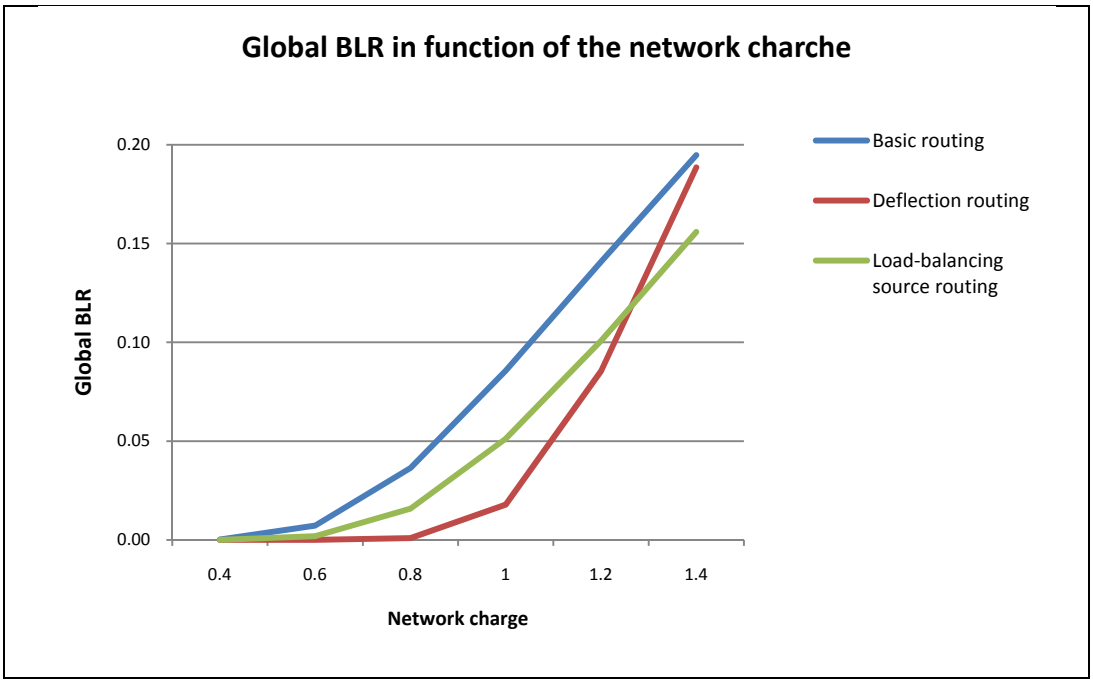


Figure 28 Global BLR in function of the network charge and the routing protocol used

For low-charged and charged networks, the contention techniques based on routing protocols (deflection and load-balancing source) improve the obtained global BLR of the network with a basic routing based on shortest path.

Concretely, the designed load-balancing source routing logic is better than the basic routing scheme in terms of BLR but it is worse than the deflection routing logic. Only for extreme-high traffic load in the network, the load-balancing source routing implemented supposes and improvement in the global BLR compared with the deflection routing logic.

If representing the BLR per route for a charged network (Fig.29), it is clearly discernible the routes that suffer burst contention with the basic routing logic and consequently have worst BLR. With de deflection

and load-balancing source routing logics the BLR in this routes is diminished, so it is corroborated that good routing logics influences positively to the burst contention in the network.

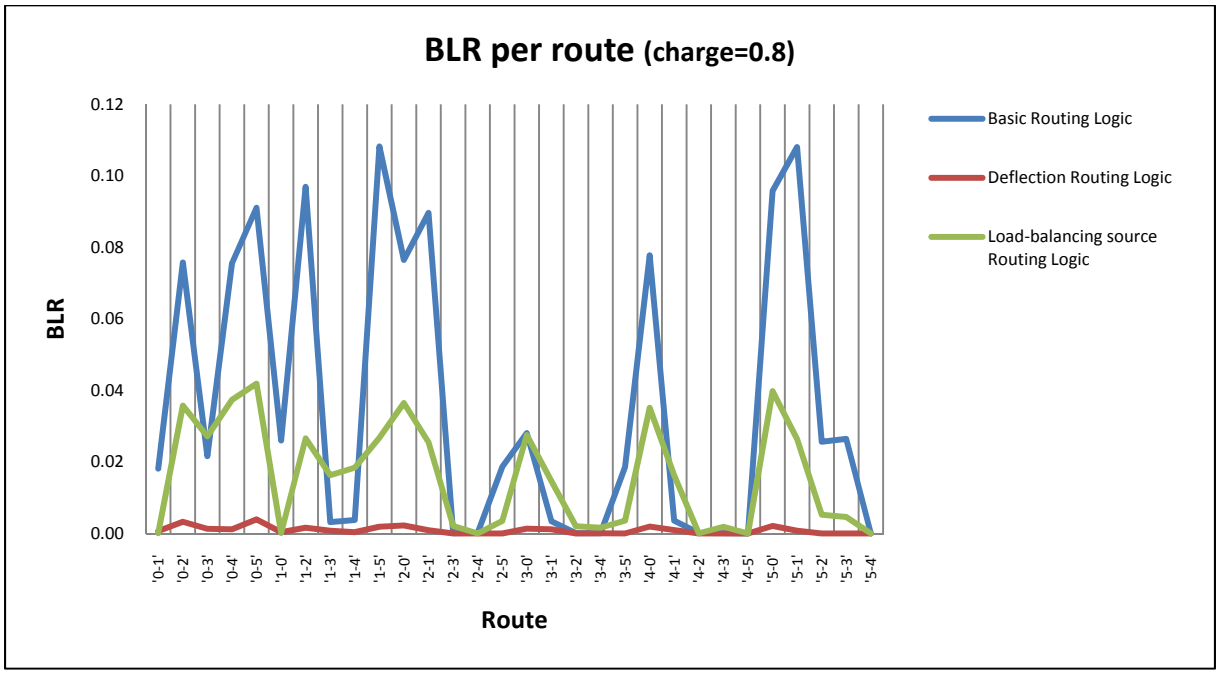


Figure 29 BLR per route in function of the routing logic (charge = 0.8)

These routes with more problems of burst contention are 1-5, 5-1, 1-2, 5-0 among other; looking the topology network graph (Fig.23), this routes uses the ports 1-0, 0-1, 0-5, 5-0, 1-2, 2-1, which effectively correspond to previous observation (Fig28) in which the ports with more lost bursts were 0-2, 2-0, 2-5,5-2,1-0 and 0-1.

One aspect to comment is the fact that the evolution of BLR per route with increasing network charge is different for each route. Therefore, for high network charges and in some routes the load-balancing source routing becomes worst as a contention resolution technique than load-balancing source or basic routing and in other routes it is the reverse.

As example, the BLR in function of the network charge for the route 0-3 and 1-0 is plotted in Fig.30.

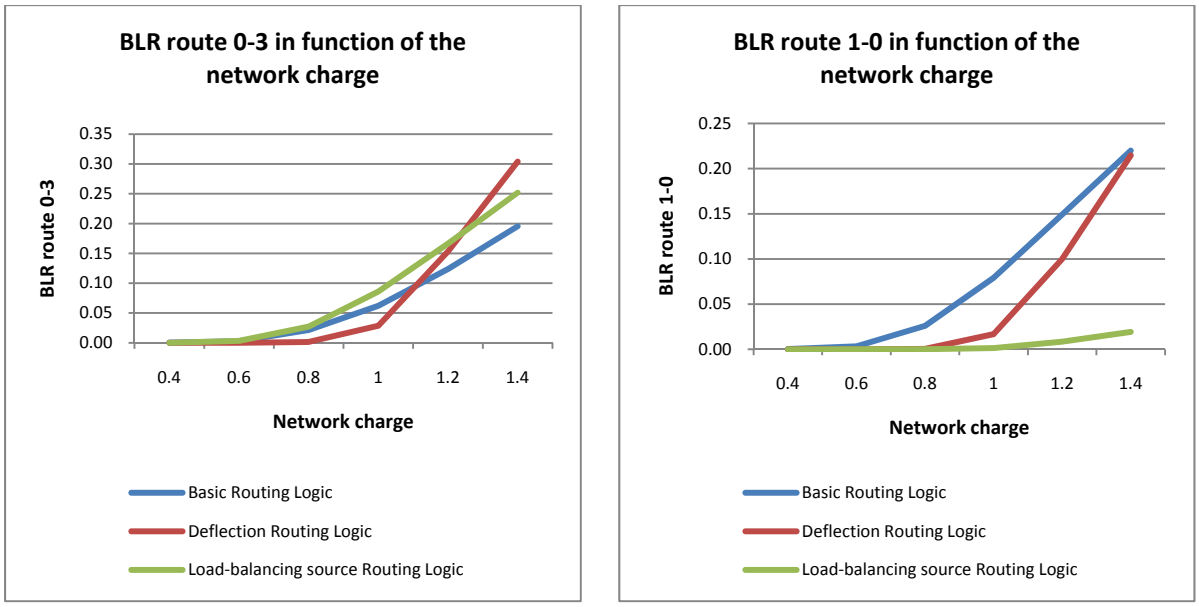


Figure 30 BLR per routes 0-3 and 1-0 in function of the network charge

With this observation, it is possible to say that routing schemes can decrease the burst contentions in some specific links better than in others and it also depends on the network charge.

3.1.4 LOAD-BALANCING SOURCE ROUTING - FUTURE WORK

The BLR of the proactive routing protocol based on load balanced-algorithm implemented has been evaluated and compared with the other routing schemes, but it is necessary to remark that the load-balancing routing scheme presented has distributed the traffic across the network using the ratios and routes introduced manually to the xml file.

As mentioned, in futures works, an algorithm for automatically calculating the best ratio to each path of the set of possible paths between a pair of nodes could be implemented given a determined network. The purpose would be to obtain optimal ratios and consequently, an optimal routes distribution of traffic to use the available resources more effectively and analyze if it can be better as a contention technique than the deflection routing in normal network charges for some situation.

Once an optimal load-balancing routing scheme with optimal ratios is defined, the next step will be to simulate with different network topologies, different switching and setup times and different scheduling algorithms in order to extract the relations and influences in the network performance of theses parameters considering the routing logic developed in this work.

3.2 VIRTUAL TOPOLOGY

The principal objective of design a virtual topology over an OBS network is to improve the network congestion and throughput metrics, using the best of both optical and electronic world [41].

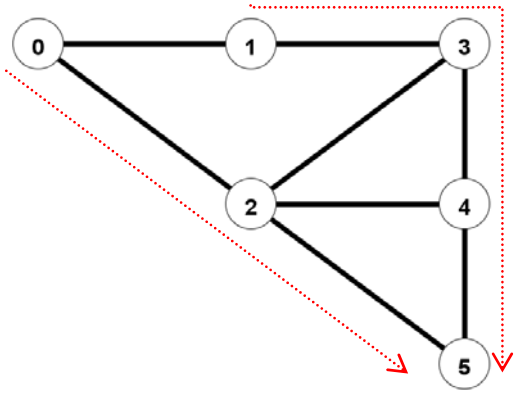
Basically, **design a virtual topology** on a physical network consists on deciding the lightpaths to be set up in terms of their source and destination nodes and doing an adequate wavelength assignment.

In this work, two virtual topology schemes are proposed:

- **Virtual Topology scheme 1:** it adapts the physical topology by adding a virtual link between the two nodes with the least degree in the network that are not connected physically. The degree of a node is the number of connections that a node has to other nodes (i.e. it is the number of different nodes that the node is connected to). This scheme tries to make the network as connected as possible.

- **Virtual Topology scheme 2:** it searches the longest path above every pair of nodes in the network and it adds the virtual link between them. This scheme focuses its attention to the shortest path with the maximum number of hops in the network (network diameter) and consequently, the route with more hops and bursts loses. The objective is to reduce the bursts lost in the longest paths.

Looking for a visual example, if the SIMPLE topology is considered and only one new virtual links is added, in the first scheme the ligthpath is established between the nodes 0 and 5, and in the second scheme, between the nodes 1 and 5:



Virtual Topology scheme 1: Virtual link between nodes 0 and 5
Lingthpath: 0 → 2 → 5

	Node 0	Node 1	Node 2	Node 3	Node 4	Node 5
Node degree	2	2	4	3	3	2

Virtual Topology scheme 2:
Network diameter = 3
Virtual link between nodes 1 and 5
Ligthpath: 1 → 3 → 4 → 5

CONNECTION	NUMBER OF HOPS
0 ↔ 1	1
0 ↔ 2	1
0 ↔ 3	2
0 ↔ 4	2
0 ↔ 5	2
1 ↔ 2	2
1 ↔ 3	1
1 ↔ 4	2
1 ↔ 5	3
2 ↔ 3	1
2 ↔ 4	1
2 ↔ 5	1
3 ↔ 4	1
3 ↔ 5	2
4 ↔ 5	1

Once the ligthpath is determined, it is necessary to assign its capacity, which corresponds to the number of wavelengths in the fiber that the virtual link will use for.

3.2.1 VIRTUAL TOPOLOGY – JAVOBS IMPLEMENTATION

To include these virtual schemes in Javobs it is necessary to work on the **topology provider** structure, which is responsible for providing the necessary information to define the network and its physical characteristics.
The idea is to adapt the simple topology (obtained from the xml file) to the virtual topology before to attach the Graph Handler and its containing network information to the experiment.

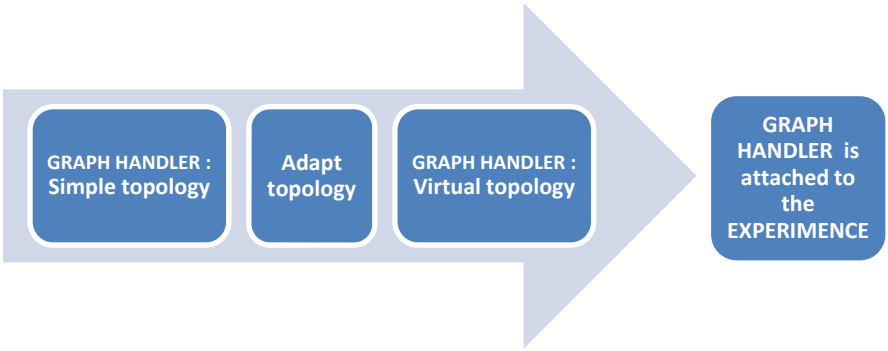


Figure 31 Virtual topology adaptation

Specifically, the adaptation of the topology to virtual topology has to be just after the creation of the experiment when the Graph Handler (agh) contains the topology network information extracted from the xml file and has to be before the experiment is ran and the experiment is conducted, because it is the moment in which the Graph Handler is attached to the experience and it cannot be changed anymore.

Basically, the topology adaptation consists on the reorganization of the number of wavelength associated to each route of the network considering the existent capacity links.
As the structure that contains information about the number of wavelength per route in the network is the **capacity matrix**. It is necessary to modify it with the purpose to considerate the new virtual links. In terms of implementation, the topology adaptation is the change of the existent uniform capacity matrix for a new nonuniform one.

If the virtual topology adaptation is not necessary because a Simple Topology provider is wanted, the adapt block does not modify anything and the information included in the definition of the network is directly the topology information that Javobs takes to define the simulation experience; which means that the capacity matrix does no change.

Then, in the case that the topology provider is a Virtual Topology provider, the main objective of the adapt-topology block is to transform the capacity matrix corresponding with to a new one that considers virtual topology links and some wavelengths are dedicated. For achieving this, it has to:

- Create an overlaid layer (to attach it to the agh) with all the physical information of the "physical layer" on it. This new LayerContainer, called "WDM layer", contains the existing physical links in the network, with its corresponding length and capacities. Each link has also a boolean variable that indicates the nature of the link (in this first step, all the variables are false, indicating that the links are physical).
- The links capacity information is captured and represented in the capacity matrix in order to be able to modify when the virtual link is added.
- Two nodes are chosen in order to create a virtual link between them. The criterion to choose these two nodes depends on the proportioned topology provider and the scheme that it implements. See the proposed schemes in the beginning of this section: Virtual Topology Provider 1 and Virtual Topology Provider 2.
- Calling the VirtualPathSetUp method, the capacities are redistributed and reassigned in the capacity matrix and a new virtual link is created in the "WDM layer" (with boolean variable in true to distinguish from the physical links). The capacity considered for the new link is introduced as a parameter selectable in the simulation experiment definition. For the capacity matrix modifications, the shortest path algorithm is applied between the nodes connected for the new virtual link in order to know the explicit lightpath and determinate the capacities in the matrix to modify. If there is no sufficient capacity in some of the segment path in the route between the pair of nodes chosen, the virtual link is not created and a message of "not sufficient capacity in links" is printed.

In the implementation done, the possibility to introduce more than one virtual link in the network is also considered. So, these two last actions have to be repeated as many times as new number of virtual links are wanted.

It is necessary to take into account the fact that the virtual link topology distributes the physical network resources (the physical capacities), but it does not append more physical links and resources.
The resulting WDM layer contains all the information about the new virtual topology network and new capacities distribution, so it is the layer attached for the simulation of the experience. The physical layer is not modified because it is interesting to preserve the information of the physical resources in the network for other possible extensions or schemes.

Virtual topology routing scheme

As the basic routing used for Javobs does not consider virtual links and cannot work with them, a basic routing scheme for simulations with virtual topology possibilities is proposed and implemented.

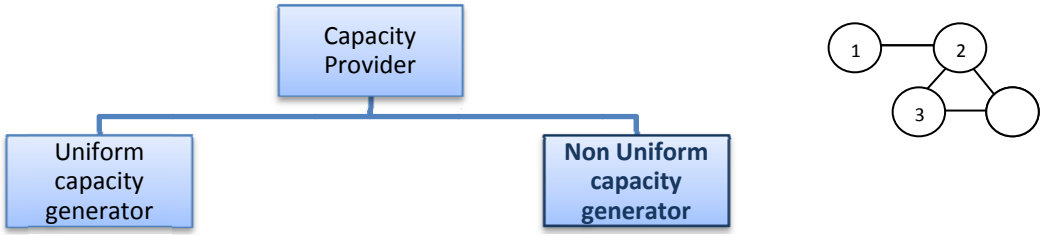
The routing scheme implemented is the adaptation of the Basic Routing Logic (which bases its routing calculation in the "physical layer" information) to virtual topology networks (which has also information in the "WDM layer").

The main difference is that given a route, if a virtual link exists between the source and destination in the "WDM layer", the getNextHop method returns the node using the virtual link in first position and then the next node that the Basic Routing logic would be returned by the shortest path algorithm. If no virtual link exists, this node is returned.

Non uniform capacity generator

For futures works on virtual topology, a non uniform capacity generator (NonUniformCapacityGenerator class) is implemented with the objective to manually introduce a matrix with a specific capacities between the nodes in the network permitting the non uniform available wavelengths in the fiber links. The purpose it would be the study of the network performance with virtual topology in these cases of non-uniformly capacity.

Then in Javobs, the capacity provider possibilities are based on the use of an uniform capacity generator with its corresponding uniform traffic matrix with capacity C and also of an non uniform capacity generator which permits to specifically introduce different capacities (C1, C2, C3,…) in different fiber links:



$$A_{NxN} = \begin{pmatrix} 0 & C & 0 & 0 \\ C & 0 & C & C \\ 0 & C & 0 & C \\ 0 & C & C & 0 \end{pmatrix} \qquad A_{NxN} = \begin{pmatrix} 0 & C1 & 0 & 0 \\ C1 & 0 & C2 & C3 \\ 0 & C2 & 0 & C4 \\ 0 & C3 & C4 & 0 \end{pmatrix}$$

3.2.2 VIRTUAL TOPOLOGY - VALIDATION

Using the link traffic detailer implemented, it is possible to evaluate the two virtual topology schemes presented and implemented in this work. The idea is to prove that the traffic between the selected nodes travels principally through the dedicated ligthpath.

In first case, considering the Virtual Topology scheme 1 and the addition of a virtual link between 0 and 5, the results of the link analyzer shows that bursts with origin 0 and destination 5 (0-5 connection in green) use the port 0-5 principally, and the port 0-2 and 2-5 secondarily in case of saturation. The port 0-5 is also used for the bursts with origin 1 and destination 5 which is the expected, because the connection 1-5 with the basic routing passes through node 0 (see Fig.32).

Considering the Virtual Topology scheme 2, and looking the number of bursts in the port 1-5, it is possible to corroborate the presence of bursts from the 1-5 connection (orange) and some bursts in ports 1-0, 0-2 and 2-5.

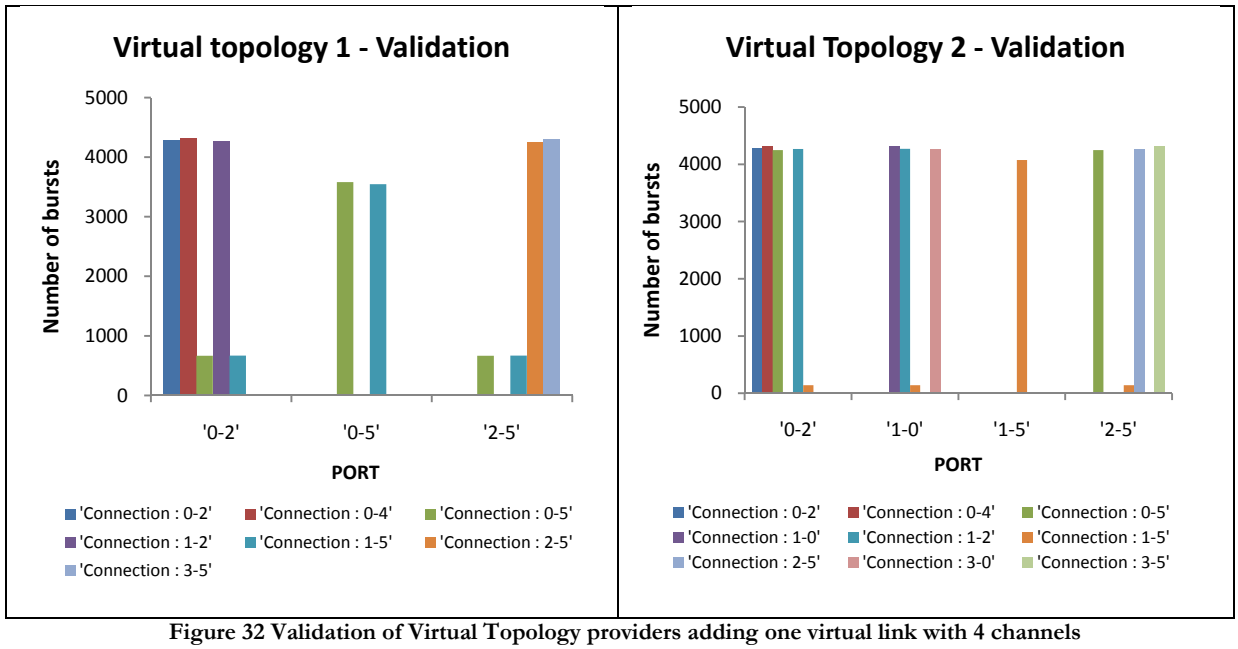


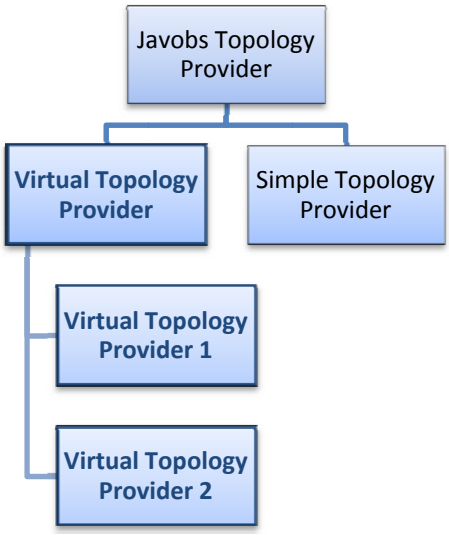
Figure 32 Validation of Virtual Topology providers adding one virtual link with 4 channels

Then, it is possible to conclude the virtual topology providers and the basic virtual routing logic are well designed and works as they were expected to.

3.2.3 VIRTUAL TOPOLOGY - SIMULATION AND RESULTS

The objective is to find the parameters that are determinant for the Virtual Topology design, and delimit the cases when the virtual topology usage supposes an improvement on the throughput and have a positive impact to the network congestion.

In order to evaluate the improvement or deterioration in performance, a comparison study is done between the different topology providers in Javobs: the simple topology provider already implemented and the two virtual topology providers proposed and implemented in this thesis.



The scenario simulated is based on the SIMPLE topology network, and general assumptions are considered (they are defined at the end of chapter 2).

For the Simple topology provider the deflection routing logic is considered, because it is better in performance than the basic routing. And for the virtual topology provider 1 and 2 the new implemented Virtual Routing Logic is used.

In Fig33 the BLR obtained for each topology provider is represented in function of the number of wavelength allocated for each one of the new ligthpath (also called capacity of the new virtual link).

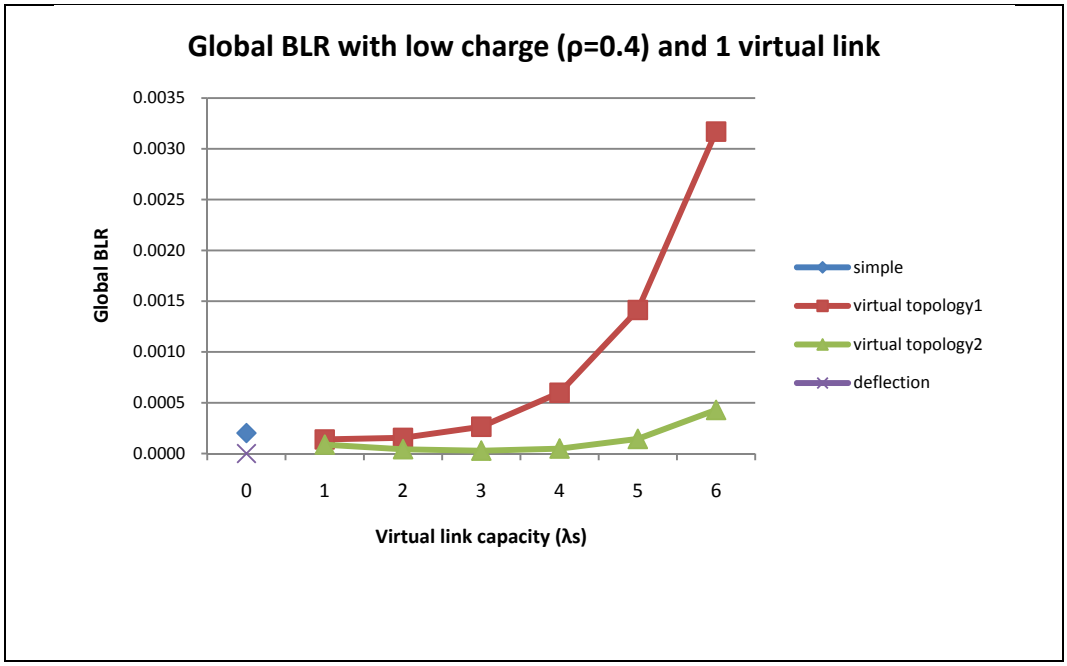


Figure 33 Global BLR with low charge (ρ=0.4) and 1 virtual link

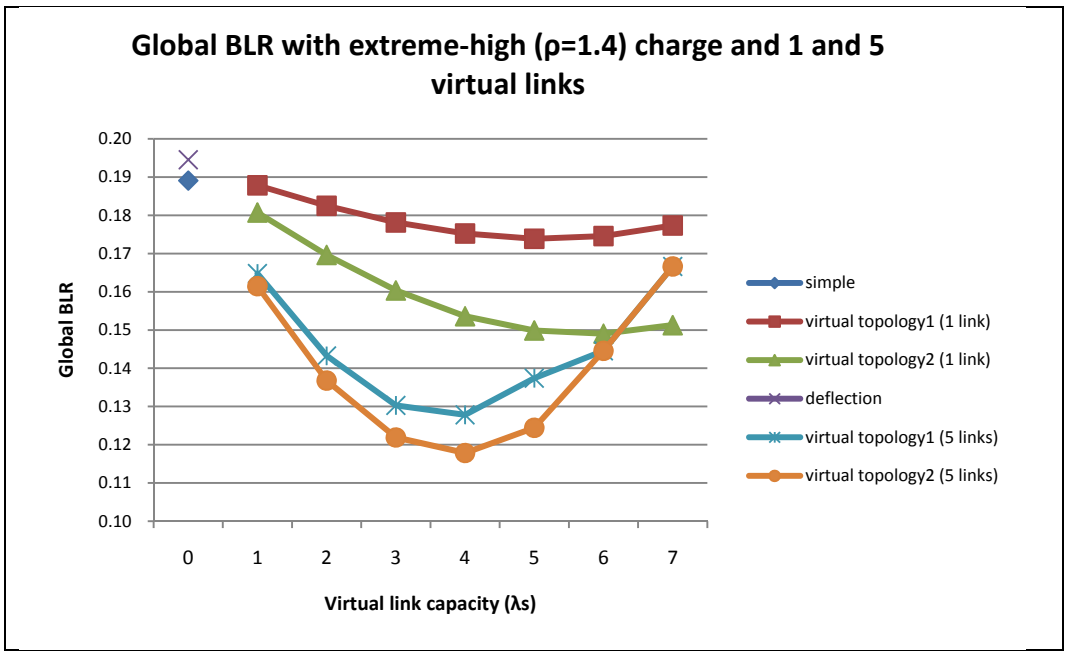


Figure 34 Global BLR with extreme-high (ρ=1.4) charge and 1 and 5 virtual links

With these results, it can be deduced that for low traffic loads, the use of virtual topology can be an improvement if basic routing logic (based on shortest path) is used, but it does not if deflection routing is used. Contrary, for extreme high traffic load, the virtual topology design supposes and improvement in the obtained BLR. This effect is due to the fact that for high traffic loads, the deflection routing does not work very well, as commented in routing section 3.2.

As it is possible to appreciate ing Fig.34, given a number of links and a virtual topology provider, exists an optimal number of wavelengths for the virtual links that optimizes the BLR in a determinate network charge.

As the BLR is determined by a specific traffic load in the network, in Fig.35, results are plotted for a range of traffic charge between ϱ=0.4 (lightly load) to ϱ=1.4 (extremely-high load):

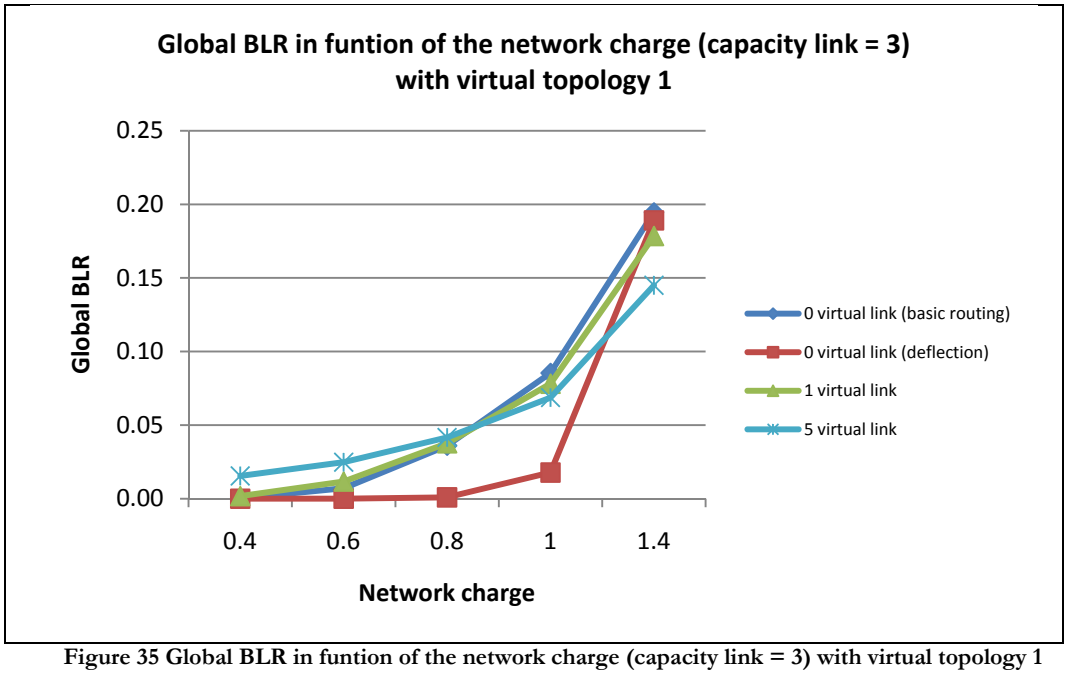


Figure 35 Global BLR in funtion of the network charge (capacity link = 3) with virtual topology 1

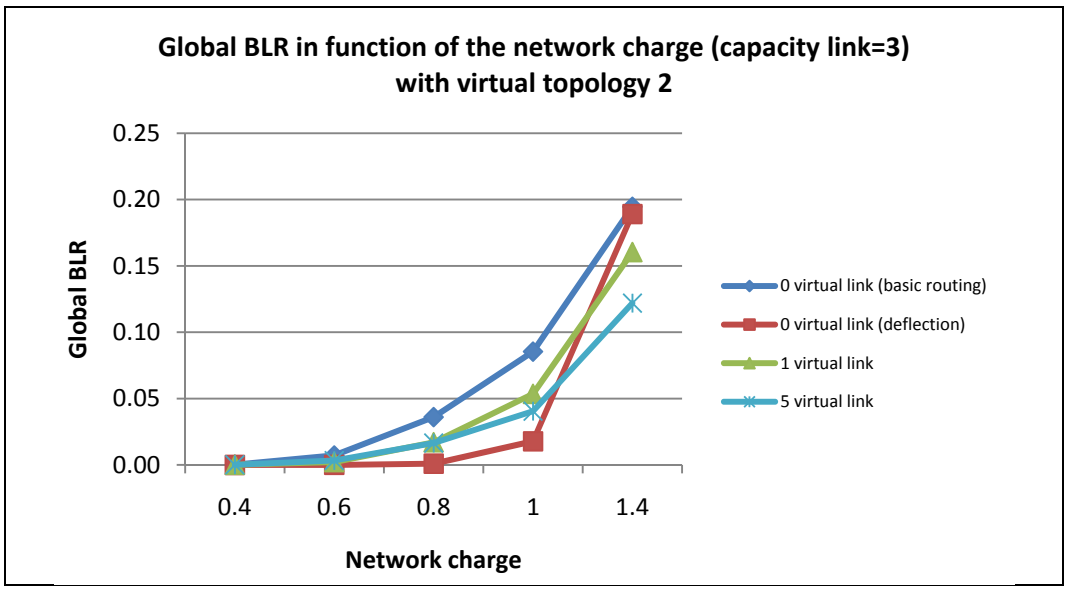


Figure 36 Global BLR in function of the network charge (capacity link=3) with virtual topology 2

There is a compromise between the amount of new virtual links that is wanted to add and the wavelengths per ligthpath that minimizes the BLR, depending on the network charge.

3.2.4 VIRTUAL TOPOLOGY - FUTURE WORK

In the virtual topology study, the challenge is to find the optimal distribution of capacities (how many wavelengths assign to the ligthpaths) and how many virtual links to add for a specific traffic load given.

In setting up a virtual topology, the considerations to take into account are the delay, the throughput, the equipment cost and reconfigurability. Basically, the principal constraint to virtual topology design is the number of wavelengths available in each fiber link.

As a future work, new methodolgy could be proposed with the purpose of finding and assigning an optimal path to the virtual link considering the congestion of the paths or the other connections done. Thus, the capacities could be well-distributed for the different routes. Actually, the basic shortest path algorithm is applied to setup the lightpaths.

Also the non Uniform Capacity generator should be used to study if there is some improvement in performance in the usage of fiber links with different capacities, i.e. the fiber links which connects the nodes in the network has different number of wavelengths available.

This study could be expanded for different topologies networks, with more nodes and also different degrees of connectivity with the purpose of deducing the topology cases in which the virtual topology introduces improvements.

3.3 TRAFFIC

3.3.1 TRAFFIC MODELING - BASIC CONCEPTS

As mentioned, in the cases considered in this work, the network traffic is mathematically represented by the assumption of a fixed burst size and an interarrival time process. This interarrival time process, in terms of modeling, is a non-negative random sequence, where each value represents the length of the separating time interval of two consecutives arrivals. In this work, the sequence of interarrival times is called and referenced as traffic trace or traffic serie.

Short-range and long-range dependent models have to be distinguished:

SHORT-RANGE traffic models: were initially used for its computational simplicity. In these models the autocorrelation function of the traffic traces decays at least as exponentially and it fluctuates around the mean.

In **LONG-RANGE traffic models**, the autocorrelation function of the traffic traces decays to zero hyperbolically and the values can be far from the mean.

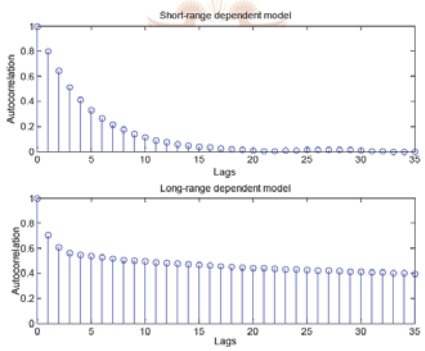


Figure 37 Short-range traffic with exponential autocorrelation and long-range traffic models with hyperbolically autocorrelation function

Concretely, for long-range traffic models the autocorrelation decays as $n^{-\beta}, as\ n \to \infty, 0 < \beta < 1$), which is an hyperbolic function rather a negative exponential (as $\rho^n, as\ n \to \infty, 0 < \rho < 1$). The slowly decaying autocorrelation is synonym of Long-Range Dependence (LRD).

LRD is a phenomenon where current observations are significantly correlated to observations that are further back in time showing a tendency to generate similar values.

Then, an interarrival time process (our traffic trace) is LRD if its current values are strongly correlated with its previous values far into the past.
As a result, it is possible to observe that the traffic trace remains at higher or lower values for relatively long periods of time and appears to exhibit trends without any cycle. In the real world, it corresponds to periods of time with high network activity when the users have a lot of network applications in execution and periods of inactivity (for example during the night).

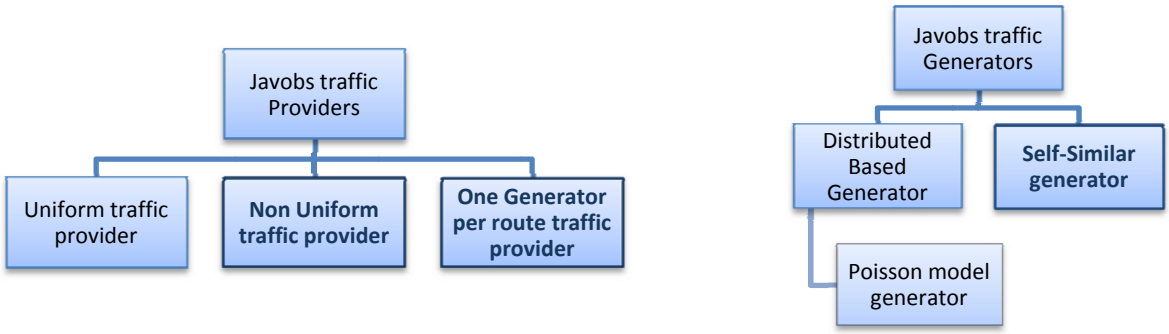
The long-range dependence degree is quantified by a single parameter called Hurst parameter, which is related to the rate of decay of the autocorrelation coefficients (β). In section 3.3.4, the Hurst parameter is treated in more detail for $0.5<H<1$ which corresponds to traffic series with LRD.

Network traffic modeling evolution

Network arrivals were (and are) often modeled as Poisson processes for analytic simplicity [32], even though a number of traffic studies have shown that packet interarrivals are not exponentially distributed [50][47][51]. In these traffic studies, it has been shown that Poisson traffic model or other simple models does not accurately reflect the traffic long-range dependence and the burstiness when viewed at varying timescales and consequently they were very optimistic in the queuing prediction.
The first experimental evidence of self-similar characteristics in Ethernet local area network traffic was presented in [47]. A subsequent investigation suggested that the same holds for wide-area network traffic [51]. In these studies they proved the exhibition of large periods of utilized links followed by large periods of silence and it was described as bursty traffic.
From there, a lot of recent work argues convincingly that network traffic is much better modeled using models with self-similar characteristics.
This chapter tries to adopt this traffic modeling evolution to Javobs simulator by the implementation of different models of traffic.

Javobs traffic model

As mentioned, in Javobs the traffic provider is the component that provides and distributes the traffic over the network. To generate this traffic, one or several traffic generators are needed.
In Javobs, the possible providers and generators of traffic are the following (the new components implemented in this thesis are represented in dark blue):



The combinations treated are:

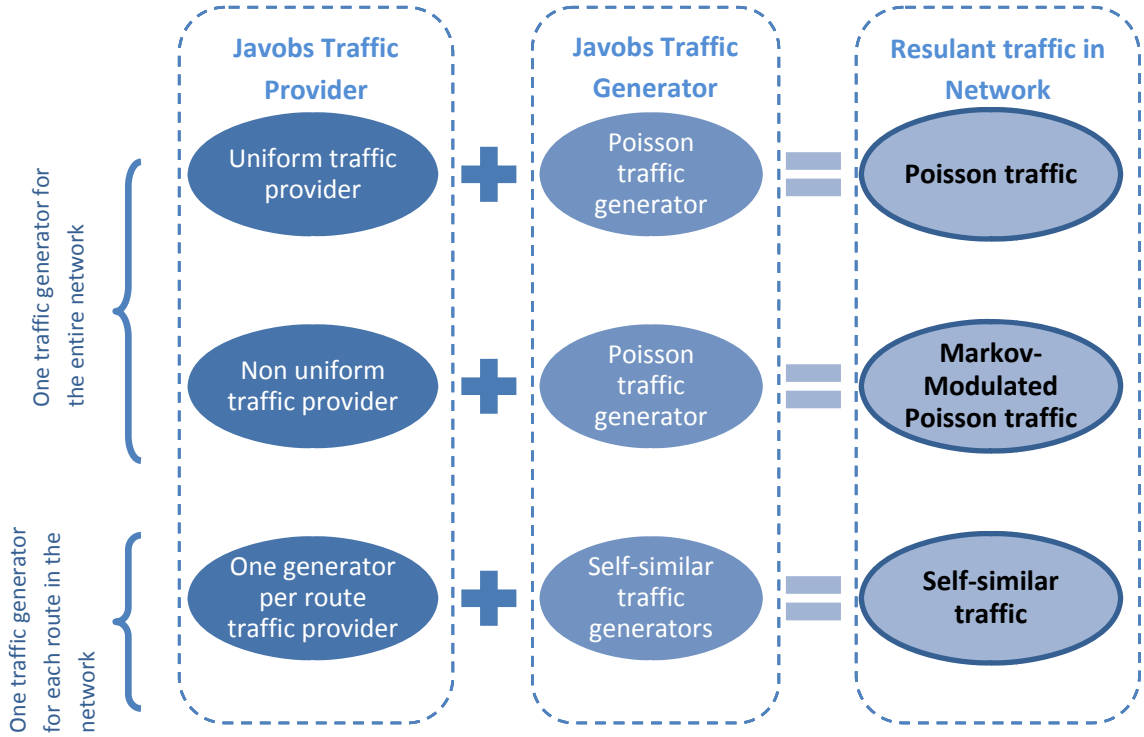


Figure 38 Javobs traffic generation models

Then, to generate Poisson traffic the UniformTrafficProvider with a PoissonTrafficGenerator are used. The NonUniformTrafficProvider together with a PoissonTrafficGenerator provides the necessary tools to generate markov-modulated traffic; and the provider called OneGeneratorPerRouteTrafficProvider with several SelfSimilarGenerators are used to obtain self-similar traffic.

Therefore, JAVOBS allows to use a unique traffic generator for all the burst in the network or to use an independent traffic generator for each edge route and edge node. In section 3.3, the need of one generator per route for the self-similar traffic model instead to use one for the entire network is justified.

The definition and implementation of the Poisson, Markov-modulated Poisson and Self similar traffic models are explained in the following sections.

3.3.2 POISSON TRAFFIC

The model to represent the number of burst arrivals in a time interval with Poisson distribution is the Poisson process. This model is the oldest traffic model in use, and was introduced in the context of telephony by Erlang [56].

A Poisson process X(t) is characterized by a rate parameter λ. For any two time instants t1 and t2>t1, X(t2)-X(t1) is the number of arrivals during the time interval (t1,t2]. The number of arrivals during this interval follows a Poisson distribution; that is,

$$P(X(t2) - X(t1) = n) = e^{-\lambda(t2-t1)} \frac{(\lambda(t2 - t1))^n}{n!}$$

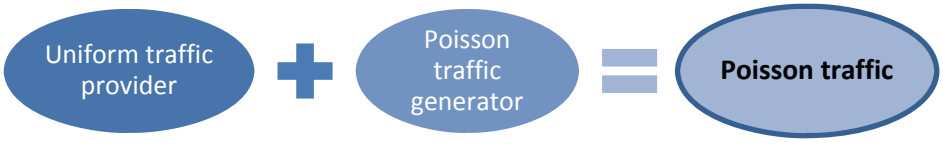
The main properties of a Poisson process are:

- Interarrival times are exponentially distributed, so the time between consecutive events follows an exponential distribution.
- The superposition of independent Poisson processes results in a new Poisson process whose rate is the sum of the component rates.
- Memoryless: the number of events in a given interval is independent of the number of events before the initial interval time, so the number of arrivals in disjoint intervals is statistically independent.

Poisson is by definition uncorrelated, so Poisson model cannot represent a long-range dependence [33]. For this reason, the principal limitation of the Poisson model is that it cannot capture traffic burstiness.

Poisson model in Javobs

As already mentioned, to generate Poisson traffic in Javobs it is necessary to use an Uniform traffic Provider, which provides the same traffic between each pair of nodes in the network together with a Poisson traffic Generator to obtain a traffic serie with exponential distribution.



As the superposition of Poisson processes results in other Poisson process, only one Poisson traffic Generator is needed to generate all the traffic that one edge node emits to all the destinations. In this manner, the offered traffic from one node is distributed between the destinations and each of the routes maintains its Poisson nature. Using the Uniform traffic Provider, the uniform distribution of the offered traffic among the destinations is assured.

3.3.3 MARKOV TRAFFIC

3.3.3.1 Markov model

The Markov modulated Poisson Process is a Poisson process whose rate varies according to a Markov chain. It is one of the simplest approaches which permit to capture some "memory" and some traffic burstiness, obtaining same behavior at some different time instants.

By definition, a Markov chain is a stochastic process where the probability of the next state only depends on the current state and is independent to the rest of past states.

The state diagram in Fig.39 shows a Markov chain with its states and their transition probabilities. The transition probabilities are the probabilities of moving from one state to another.

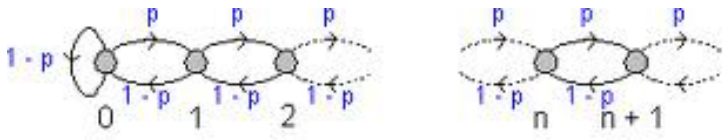


Figure 39 General Markov chain

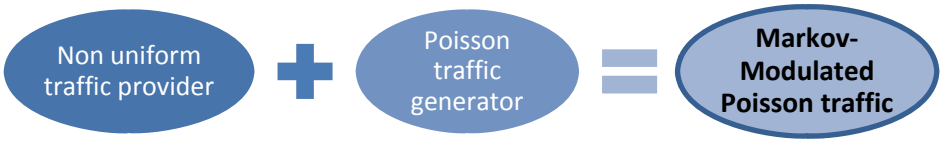
Markov chains combined with Poisson process results in the Markov-modulated Poisson process. Each state of the Markov process represents a different probability distribution for the Poisson process. Therefore, when the state of the Markov chain changes, the rate of the Poisson process also changes. Thus, the state of Markov process controls the probability distribution of the Poisson traffic trace.

The Poisson process is modulated in such a way that the arrival rate at any time depends on the state of the Markov process. This process is completely defined by a transition matrix and eventually an initial state.

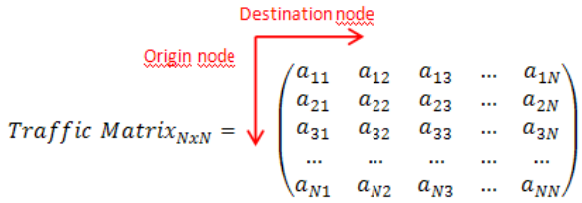
Contrary to Poisson, the Markov process introduces some dependence in interarrivals by the addition of some non zero autocorrelation in the trace. So, Markov processes are memoryless systems but they have limited memory of the past, because of the future state only depends on the actual one.
Therefore, they can reflect short-range dependence, but they cannot reflect long-range dependence.

3.3.3.2 Markov-traffic - Javobs Implementation

To obtain a Markov-modulated Poisson traffic generator, a Poisson traffic generator is used and a new traffic provider is developed considering the possibility that the different routes in the network have different traffic loads and also that these route traffic load changes over the time following a Markov process.



This new traffic provider is called NonUniformTrafficProvider in this work and as all the traffic providers it has a structure that includes and preserves the information for the traffic charge in each of the routes in he network. This structure is the **traffic matrix**, which is a matrix NxN, with N the number of nodes in the network.



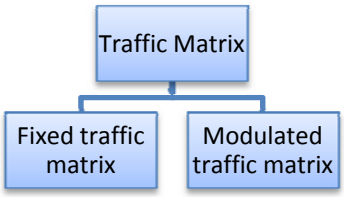


Figure 40 Traffic Matrix possibilities

For convention, ech element of the traffic matrix is the traffic charge that node *i* emits to node *j*.

- If is the same for any *i* and *j* → Uniform traffic matrix → the traffic charge is the same independently of the route. Example:

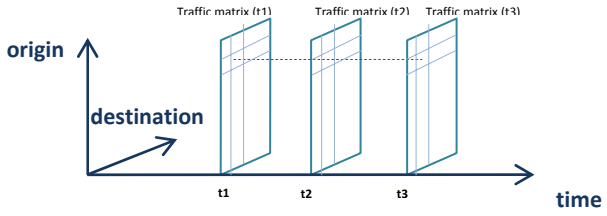
$$Uniform\ Traffic\ Matrix = \begin{pmatrix} 0.4 & 0.4 & 0.4 \\ 0.4 & 0.4 & 0.4 \\ 0.4 & 0.4 & 0.4 \end{pmatrix}$$

- If each is fix and does not change along the time → Non Uniform-Fixed traffic matrix → each route has a specific charge of traffic. Example

$$Non\ Uniform\ Fixed\ Traffic\ Matrix(t) = \begin{pmatrix} 0.4 & 0.6 & 0.5 \\ 0.6 & 0.3 & 0.4 \\ 0.5 & 0.4 & 0.7 \end{pmatrix}$$

- If each changes along the time → Non Uniform - Modulated traffic matrix → each route has a specific charge of traffic which varies along the time.
  - Example: *L(t)= {0.4, 0.5, 0.5, 0.4, 0.7, 0.6..}*

$$Non\ Uniform\ Modulated\ Traffic\ Matrix(t) = \begin{pmatrix} L(t) & L(t) & L(t) \\ L(t) & L(t) & L(t) \\ L(t) & L(t) & L(t) \end{pmatrix}$$



In this section, the non-uniform modulated traffic matrix is used.

As the number of nodesN is unknown before the simulation starts, a TrafficMatrixCreator is needed to create the traffic matrix once the experiment is running and the number of nodes in the network is known.

As it is necessary that each element evolutes along the time following a Markov process L(t), a MarkovProcess component is implemented. In order to create each one of the Markov processes for each one of the routes in the network, a Markov process creator is also implemented. Then, it is necessary to create as number of MarkovProcesses as number of elements in the Traffic Matrix is (NxN).

The different values in the Markov process L(t) are introduced during the model construction and they represent the possible traffic loads that the source emits to its destinations. If L(t) values are {0.4, 0.5, 0.6}, means that the traffic load between each pair of nodes in the network varies between 0.4, 0.5 and 0.6 as time progresses.

For to achieve the time progression, the MarkovProcess component has to be able to actualize load values when the traffic matrix is updated.

The way the load values change depends on the Transition Matrix. The Transition Matrix defines the probability to change the traffic load for another determinate load considering the following Markov chain:

New state(j)

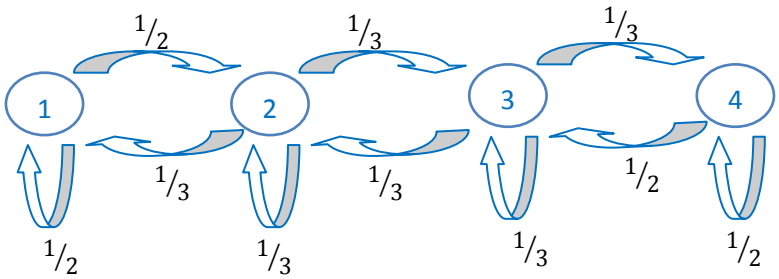
Past state (i)

$$Transition\ Matrix_{NxN} = \begin{pmatrix} p_{11} & p_{12} & p_{13} & \cdots & p_{1N} \\ p_{21} & p_{22} & p_{23} & \cdots & p_{2N} \\ p_{31} & p_{32} & p_{33} & \cdots & p_{3N} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ p_{N1} & p_{N2} & p_{N3} & \cdots & p_{NN} \end{pmatrix}$$

For each state of the Markov chain, the sum of its transition probabilities is 1:

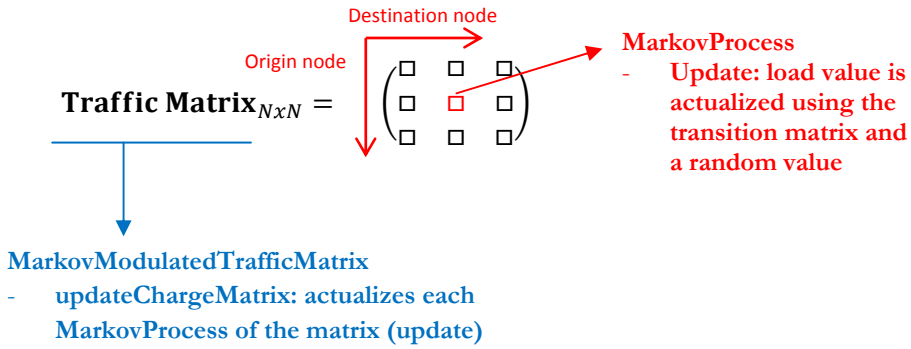
$$\sum_{i=fix}^{N} pij = 1$$

For this work, the following Markov chain and transition matrix is considered:



$$Transition\ Matrix_{NxN} = \begin{pmatrix} 1/2 & 1/2 & 0 & 0 \\ 1/3 & 1/3 & 1/3 & 0 \\ 0 & 1/3 & 1/3 & 1/3 \\ 0 & 0 & 1/2 & 1/2 \end{pmatrix}$$

The actualization of the Traffic Matrix (updateChargeMatrix method) and consequently the actualization of each one of the Markov process (update), is done at the end of each step in the simulation. Then, when each edge node generates traffic in the following step, it already has the actualized values of the traffic charge associated to each route.



Therefore, the non-uniform traffic provider generates traffic the Poisson which is distributed over the network depending on the Traffic Matrix described, which contains information of the specific charge emitted to each route in a determinate moment.

3.3.3.3 Markov traffic - Validation

To achieving this, the PerRouteAnalyser is used and a simulation with repetitions is done in order to obtain more accurate results with the simulation. For the traffic observation, which changes over the time it is interesting to consider more than one sample of results, i.e more than one repetition (Fig.41)

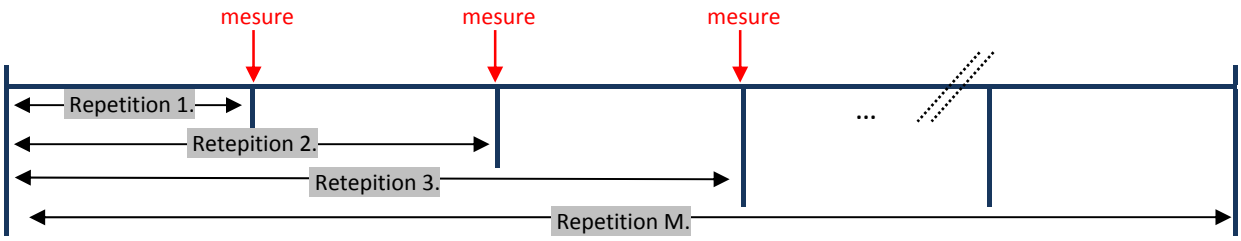


Figure 41 Simulation with repetitions for the validation of Markov-modulated Poisson traffic

In order to validate that the NonUniformTrafficProvider is capable to emit different traffic load to each different route in the network and that this traffic load corresponds with the value in the Traffic Matrix it is necessary to simulate for a known non uniform Traffic Matrix, i.e a non uniform fixed traffic matrix. For example the following:

$$Fixed\ traffic\ matrix_{6x6} = \begin{pmatrix} 0.0 & 0.2 & 0.3 & 0.1 & 0.2 & 0.2 \\ 0.1 & 0.0 & 0.3 & 0.4 & 0.1 & 0.1 \\ 0.1 & 0.2 & 0.0 & 0.4 & 0.2 & 0.1 \\ 0.1 & 0.2 & 0.3 & 0.0 & 0.2 & 0.2 \\ 0.1 & 0.2 & 0.3 & 0.2 & 0.0 & 0.2 \\ 0.1 & 0.2 & 0.3 & 0.1 & 0.3 & 0.0 \end{pmatrix}$$

For example, the calculated offered traffic emitted from node 0 (which corresponds to the first row in the matrix) is:

	Node destination 1 ($\rho = 0.2$)	**Node destination 2 ($\rho = 0.3$)**	**Node destination 3 ($\rho = 0.1$)**	**Node destination 4 ($\rho = 0.2$)**	**Node destination 5 ($\rho = 0.2$)**
Calculated offered traffic per route (origin 0)	2.3	3,5	1,152	2,3	2,3

These values effectively correspond to the obtained results for the analyzer in Fig.42:

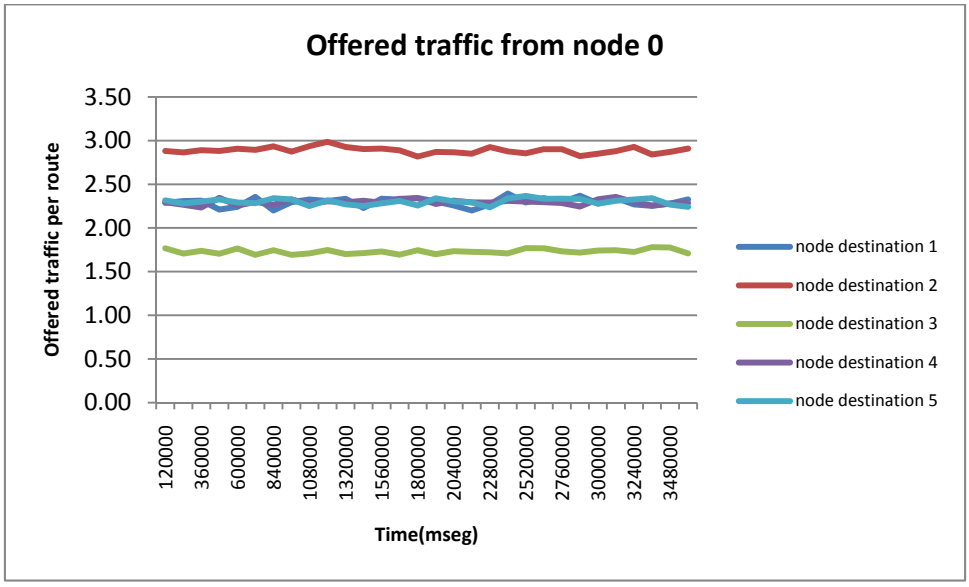


Figure 42 Offered traffic from node 0

Then, it is possible to affirm that the Non Uniform traffic provider distributes correctly the traffic loads of the traffic matrix. Next step, is to prove how it works if each element of the traffic matrix changes over the time, i.e. if the traffic matrix is a Non Uniform Modulated Traffic Matrix.

In Fig.43 a representation of the offered traffic in function of the time for one specific route (route 0-2) in the network is showed:

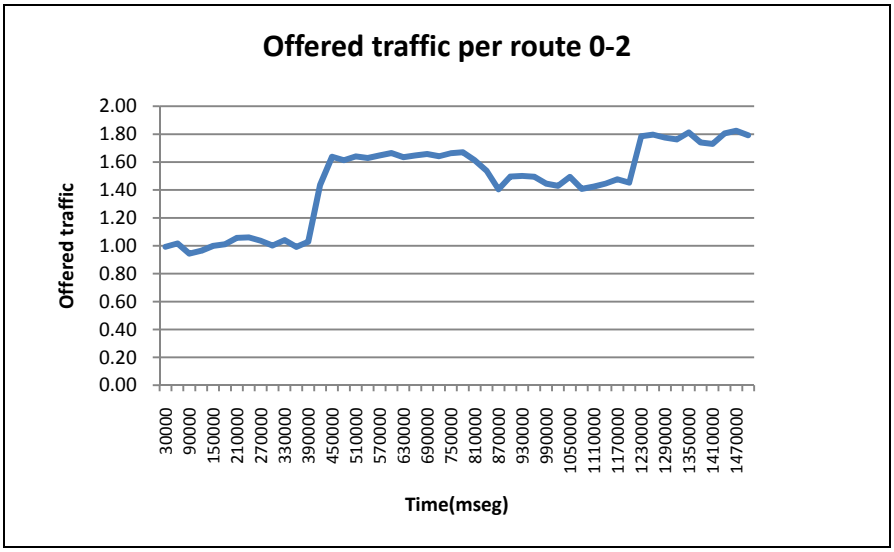


Figure 43 Offered traffic from node 0

It is clearly observable the moment in which the traffic matrix is actualized, which corresponds to a change of stat in Markov chain. It is possible to observe the different levels of traffic load.

Then, it is validated that the traffic distributed for each route corresponds with the traffic charge coefficient included in the Non Uniform Modulated Traffic Matrix and also it follows a Poisson process in each stat.

With this two proves, a markov-modulated Poisson traffic has been obtained and validated.

3.3.4 SELF-SIMILAR TRAFFIC

3.3.4.1 Self-similar model

Self-similarity is a statistical property, which can be partially characterized by a heavy-tailed probability distribution. The mathematical definition self-similar process can be found in [47-51] and its most important features are:

- The autocorrelations decay hyperbolically rather than exponentially fast.
- Non-summable autocorrelation function <=> long-range dependence <=> Self-similar traffic exhibits dependencies over a long range of time scales.
- Slowly decaying variances
- Aggregating streams of self-similar traffic typically intensifies the self-similarity ("burstiness") rather than smoothing it

As already mentioned, the data in the real networks are characterized by high or extreme variability [47,48].

Large variability in space and time generally causes that some statistical properties are repeated themselves at many time scales. This is an evidence of **fractal behavior**, in which independently of the time or space scale used it is possible to see similar patterns and "resemblance" on traffic.

Hurst parameter

The Hurst parameter describes the self-similarity degree in self-similar models, by measuring the degree of long-range dependence existent in the time-serie. Its value lies to $0.5 < H < 1$ for self-similar traffic series. This parameter cannot be calculated exactly because it would be necessary to work with infinite time-series, but different methods to estimate it are used. An overview of a large number of these estimation methods can be found in [53, [54], [55].

It is important to know that there is no best estimator to use, because each kind of estimator has its advantages and drawbacks depending on the case and they often produce conflicting results in the Hurst estimation due to the asymptotic nature of the Hurst exponent [53].

3.3.4.2 Self-similar traffic generation – Generator Implementation

The main objective is to obtain real-time-series with certain degree of self-similarity in order to use as interarrival time-serie in the traffic generation process.

As explained in [52], it is possible to obtain self-similar characteristics from the following mathematical models:

- Exactly self-similar Fractional Gaussian Noise (FGN), which its applicability is restricted
- The asymptotically self-similar F-ARIMA (fractionally differenced autoregressive integrated moving average models), in which the running time to generate a time-serie of length n is proportional to n^2.
- An M/G/∞ queue where arrivals are given by a Poisson process and have service times drawn from a heavy-tailed distribution with infinite variance. The calculation speed decreases with the number of nodes in the network.
- Method based on the Fast Fourier Transform (FFT) to generate a serie of complex numbers corresponding to the power-spectrum of FGN. This is the fastest methods for computing self-similar samples.
- Aggregate multiple sources of Pareto-distributed ON and OFF periods. In which the objective is multiplexing ON/OFF sources that have a fixed rate in the ON periods and its principal characteristic is that ON/OFF period lengths are heavy-tailed.

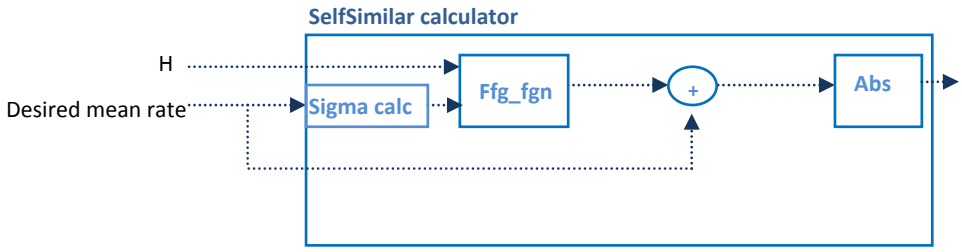
In this work, the fast Fourier transform method for generating an approximation for Fractional Gaussian Noise (FGN) proposed by Paxson is used [48]. This method is shown to be as fast or faster than other methods of generating self-similar time-series, so it is chosen for its quickly computation.

The Paxon's algorithm is based on synthesizing time-series that have the same power spectrum as FGN.

Using the Paxson's generator fft_fgn [48, 49], the authors of [35] generate synthetic self-similar interarrival times for discrete-event simulations, synthesizing a self-similar Fractional Gaussian Noise (FGN) process.

Applying the same idea, a generator fft_fgn for Javanco platform is adapted and implemented in Java launguage. SelfSimilarCalculator component includes the fft_fgn generator and some needed transformations. These transformations are motivated by the necessity to control the trace mean (which corresponds to the resultant traffic mean) and its standard deviation and also to obtain a positive-real-value-series, because as it is known an interarrival time or corresponding rate cannot be negative.

The system associated to the SelfSimilarCalculator to generate self-similar traces is the following:



The input parameters to SelfSimilarCalculator are the desired Hurst parameter that defines the degree of self-similarity and the desired mean rate to control the rate of traffic generated.

The relationship between the rate of traffic and the interarrival times-series obtained by the SelfSimilarCalculator is:

$$rate = \frac{burst\ size}{interarrival\ time}$$

Another parameter to consider is the standard deviation (or sigma) of the time-serie, which corresponds to the square root of the variance, and defines the variability or dispersion around the trace mean. The standard deviation is adjusted with the purpose to obtain a traffic trace with high rate variability and with rate values between 0 and two times the mean rate.

In Fig 44, a schematic example of the sigma adjust is presented.

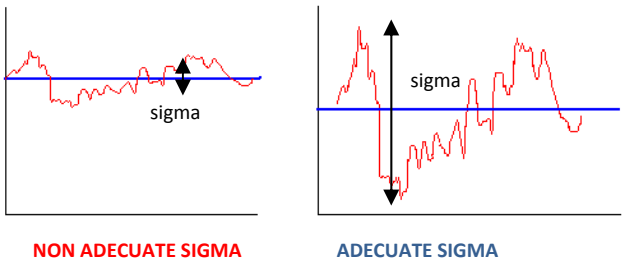


Figure 44 A non-adequate sigma adjust and an adequate sigma adjust

Ideally, the standard deviation (sigma) would be near to the desired mean rate but with an amount of negative values in the trace as minor as possible. To achieving this, it is computed from the desired mean rate as:

$$standard\ deviation = desiredMeanRate - factor * desiredMeanRate$$

Once the fft_fgn gives the resulting time-serie using the H and sigma introduced, as the resulting trace has null mean, it is necessary to add to all the values in the time-serie the desired mean rate. Then, for the remaining negative values, an absolute operation is done.

If an adequate sigma is used, the lost of mean caused for the absolute operation is minimum. The *factor* value is adjustable, but for 0.6 the results are already good for accomplish the high variability condition presented in Fig 44.

So, the obtained time-serie for the SelfSimilarCalculator corresponds to a rate trace. Then, the interarrival time-serie is inversely proportional to it multiplied for the burst size.

3.3.4.3 Self-similar traffic generation – validation

To validate that the traffic generated has self-similar properties it is necessary to estimate the Hurst parameter of the interarrival time trace generated.

In order to test the SelfSimilarCalculator class and to corroborate that the obtained time-serie have approximately the desired mean rate and the desired Hurst parameter a test tool is implemented in mainTestSelfSimilar class.

Its functionality is basically to generate different time-series with different desired input parameters (desired mean rate and desired Hurst parameter) and evaluates the corresponding obtained parameters through a visual inspection in a displayer graphics.

In the Hurst parameter estimators study realized in [53] it is concluded that when the time-series are generated by fractional Gaussian noise (FGN), the Whittle and Periodogram estimators obtain the most accurate estimation.
In addition, the Whittle estimator is considered the most robust in a lot of studies, for these reasons it is used in this work to capture the self-similarity degree.

A Whittle estimator provided by Selfis Tool, presented in [53] and created with the objective to identify LRD in a time-serie given is used for the Hurst parameter estimation.

The SelfSimilarCalculator is proved (mainTestSelfSimilar) considering different seeds, different desired rates and different values for the desired Hurst parameter. The results obtained on average are:

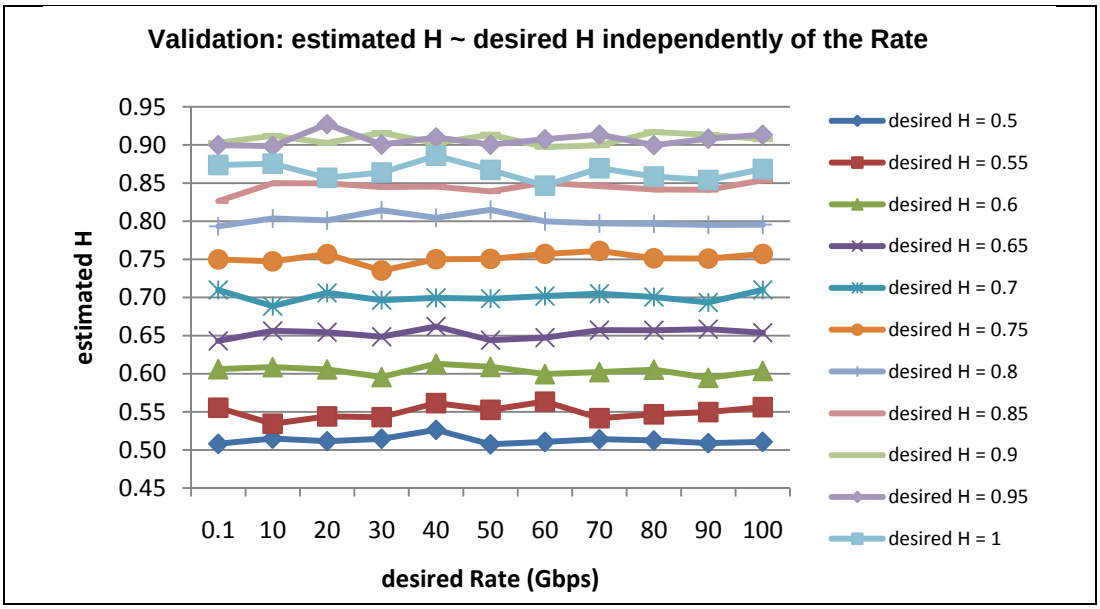


Figure 45 Validation: estimated H ~ desired H independently of the Rate

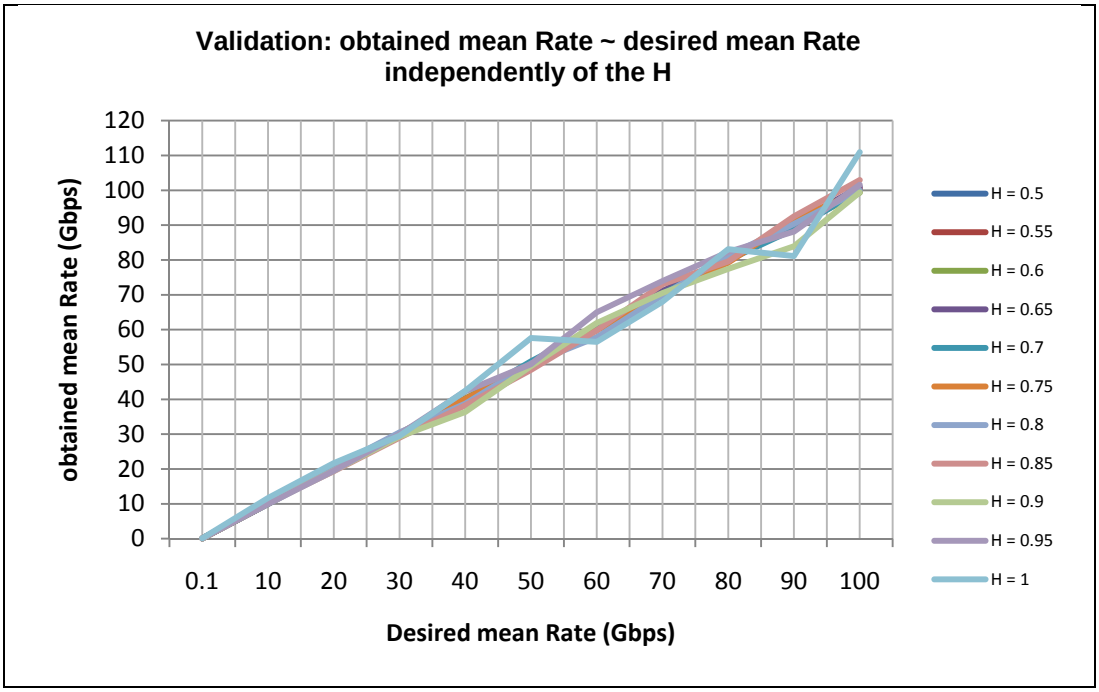


Figure 46 Validation: obtained mean Rate ~ desired mean Rate independently of the H

Looking at Fig.45, it is shown that for any desired mean rate, the estimated Hurst parameter computed from the obtained time-series is approximately the same as the desired one, with the exception of Hurst parameter values higher than 0.9 ($H > 0.9$); in which cases, it is quite difficult to conclude anything precise. Then, with SelfSimilarCalculator the Hurst parameter is reproduced for values between 0.5 and 0.9 ($0.5 < H < 0.9$).

Also the obtained mean rate is represented in function of the desired mean rate. It is possible to remark the linearity in the graph; so obtained mean rate is the same as desired mean rate, independently of the Hurst parameter (Fig.46).

With this validations, the self-similar time-serie generated by SelfSimilarCalculator is thus proved to show the Hurst parameter and the mean rate that it is expected to.

As additional comprobation, the autocorrelation function of one trace generated with the SelfSimilarCalculator presented is also represented in Fig.47 in order to visually prove its hyperbolically distribution.

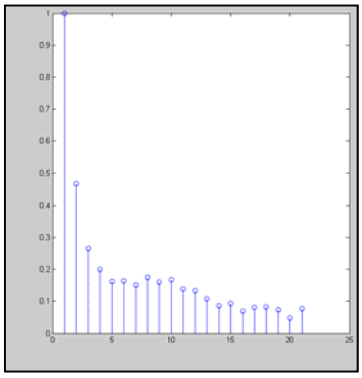


Figure 47 Autocorrelation function of one trace generated with the Self-similar generator (Matlab calculated)

Effectively, the autocorrelation of the trace decays hyperbolically to zero, but never reachs zero; so the resulting trace is long-range dependent as it was expected to.

An expanded study in the statistics of the traffic generated with the self-similar generator implemented could be done to evaluate and deepen in theoretical aspects (LRD, heavy-tailed distribution, variances, …)

3.3.4.4 Self-similar traffic – Insertion to Javobs simulator

In the case of Poisson and Markov Modulated Poisson traffic models, one stationary traffic source it is used to generate all the traffic in the network. This traffic can be distributed for the different edge nodes and for the different routes, due to the property of Poisson process which says that the sum of stationary traffic is also stationary traffic.
This is not the case for self-similar traffic, so it is necessary to use one self-similar traffic source per route, which in large networks would imply high computational cost in simulation.

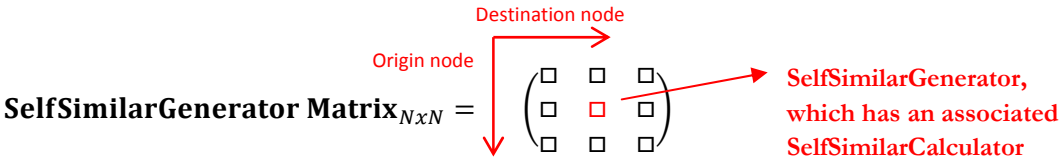
So, a new traffic provider which has one generator per route is implemented (**OneGeneratorPerRouteTrafficProvider**). This new traffic provider creates, for each route in the network, a self-similar generator called SelfSimilarGenerator, which use the SelfSimilarCalculator presented and validated before for each route in the network.

For achieving this, a creator of SelfSimilarGenerator (SelfSimilarGeneratorCreator) is implemented and its basic function is to create a SelfSimilarGenerator when required.

The principal functions of the SelfSimilarGenerator are:

- To assign the desired mean rate and create the SelfSimilarCalculator with the input parameters required: the Hurst parameter and the desired mean rate (setNominalRate method).
- To give the following interarrival value calculated by the division between the burst size and the following output value of the SelfSimilarCalculator system.

In general terms, the execution order of the traffic generation process is the following:
When Javobs Model Builder manages the Experiment, the configuration of the Traffic Provider is done. This configuration includes the creation of a Self-Similar Generator matrix corresponding to each traffic route generator and the creation of a Self Similar Calculator for each SelfSimilarGenerator assigning the desired rate mean (setNominalRate method).



Once the simulation has started, for each edge and for each route, in each step the traffic is generated. This traffic is generated for the corresponding SelfSimilarGenerator (depending on the route), and it corresponds to the following interval time given by the SelfSimilarCalculator.

3.3.4.5 Self-similar traffic – Validation

The first thing to test is whether the traffic generated for one route corresponds to the Self-similar traffic which is expected to. The expected offered traffic per route is:

$$Total\ offered\ traffic\ per\ route\ (erlangs) = 0.72$$

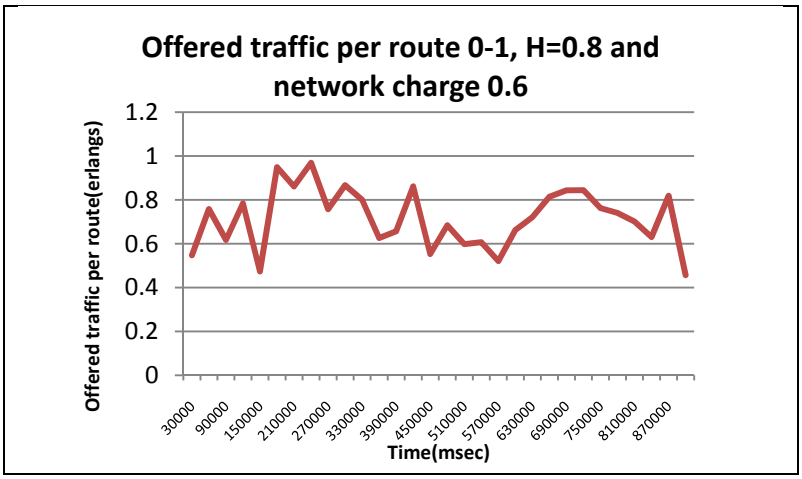


Figure 48 Offered traffic per route 0-1, H=0.8 and network charge 0.6

Observing the self-similar trace over the time (Fig.48), it is possible to observe the burstiness of the traffic, which implies periods of less activity followed by periods of more activity. The traffic mean rate approximately is 0.7, corresponding with the value obtained before.

As it is known, the degree of self-similarity typically depends on the utilization level of the network and it increases as the utilization increases.

Looking the BLR per route obtained for different Hurst parameters in function of the network charge showed in Fig49, it is possible to corroborate that the network performance degrades gradually with increasing self-similarity, but not so strictly as it is expected to (see for example, the BLR for H=0.7 for high traffic load ϱ=1 or ϱ=1.2).

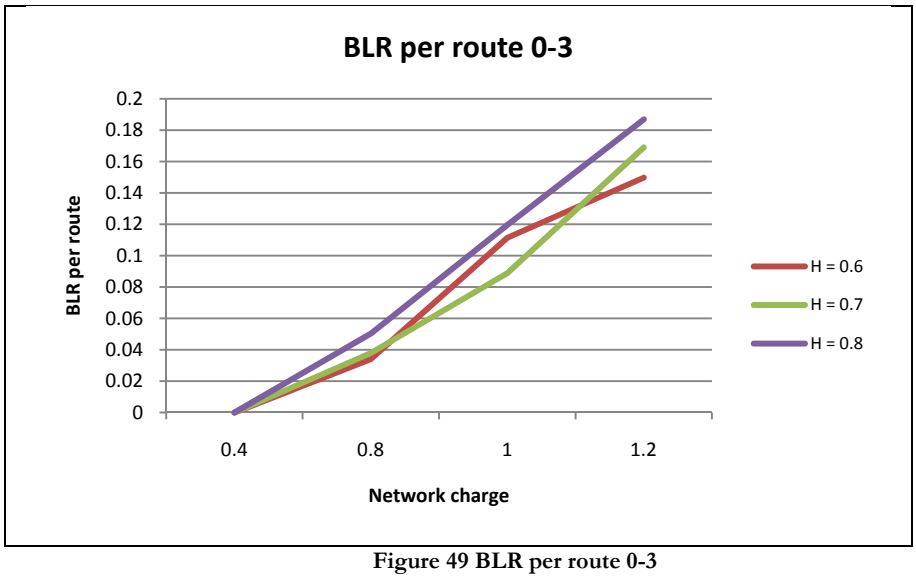


Figure 49 BLR per route 0-3

With the validation of the SelfSimilarCalculator and after the self-similar traffic generator is possible to affirm that self-similar traffic with certain degree of self-similarity can be used in Javobs simulator.

3.3.5 TRAFFIC PROVIDERS: POISSON + MARKOV-MODULATED POISSON + SELF-SIMILAR

With Javobs simulation results a comparison of the network behavior using Poisson, Markov-modulated Poisson and self-similar models of traffic can be done.

The principal features which have been validated are the following:

- The Poisson model has no memory and it is "constant" along the time, which as mentioned in real data it is not true.
- The Markov-modulated Poisson traffic changes the traffic load after determinate period of time in each route permitting different levels of traffic load, but it has limited memory.
- The self-similar traffic is the traffic model which captures better the burstiness, depending on the Hurst parameter.

The Fig.51 shows the BLR for the route 0-3 in function of the Poisson and Self-similar traffic providers:

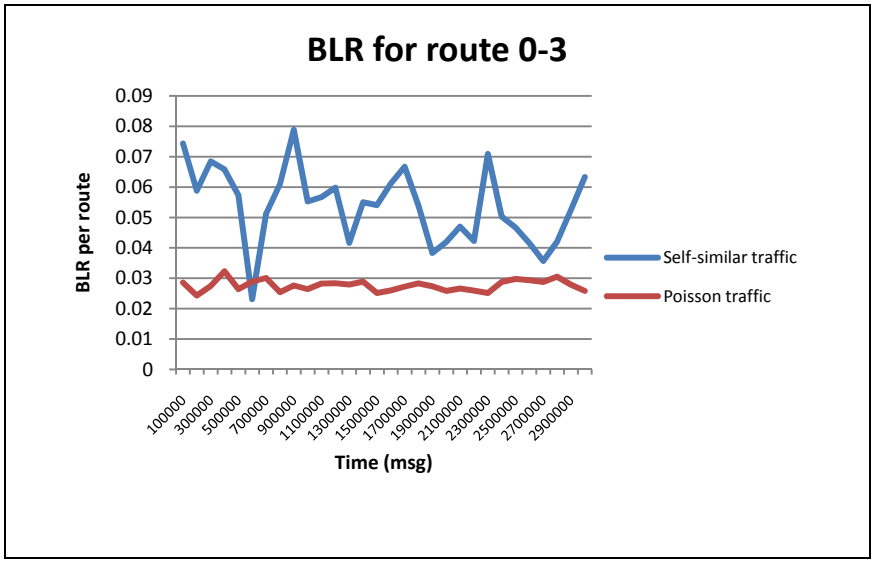


Figure 50 BLR for route 0-3

As it was expected to, the BLR is sensible to the traffic model used.

It is important to remark that the assumed traffic model for simulators has to reflect as much as possible the relevant characteristics of the traffic it is supposed to represent in order to analyze how reacts and change the network performance given a particular network conditions (for example congestion).

3.3.6 TRAFFIC - FUTURE WORK

Once the traffic generator is implemented and considering that it keeps on the desired self-similar characteristics wanted, it is moment to use it in simulations with the purpose to look for new congestion control schemes and to intensify the studies in the existent and established protocols. Most of these established schemes were thought for the past models (e.g Poisson traffic) and nowadays, they are no entirely valid and satisfactory because of the self-similarity evidence measured in network traffic. So, for future works, the re-study of these protocols with the self-similar traffic model implemented is very interesting in order to consider more realistic results.

3.4 General conclusions and future work

Simulators are essential to evaluate the network behavior in certain circumstances and how this circumstances influence in the network performance.

In this thesis, the Javobs simulator has been described and some new possibilities have been developed and validated.

Focusing in the burst contention problem, a new proactive routing protocol has been proposed with the purpose to be able to compare with the reactive routing protocol (deflection routing) already implemented in Javobs. This proactive routing protocol implemented is based on a Load-balancing source routing scheme, which can distribute the traffic over the network by splitting the traffic load along several different paths.
One point to remark is that the new routing scheme implemented is static, i.e. once the burst route is fixed in the origin, it cannot be changed; which implies that it has certain limitations as a contention resolution technique comparing with the deflection routing technique.
It has been proved that the deflection and load-balancing source routing schemes reduce the burst contention, being the load-balanced source routing protocol better for extreme high traffic loads but not for low and normal charged networks.
For future studies, it would be interesting to work on an algorithm to compute an optimal distribution of bursts over the paths.

Following with the objective to reduce the burst contention, the virtual topology topic has been treated.
The virtual topology design is based on the choice of the nodes in which the establishment of explicit ligthpaths has to be done and it also based on a correct wavelengths allocation in the network links.
Therefore, the key aspect on virtual topology is to determine and allocate from the entire set of wavelengths available in the link, a subset of wavelengths that will be used for one o more virtual links.
It has been seen that there are an optimal number of wavelengths to assign to the virtual link and it depends on the network charge.
The principal remark extracted from the simulations done is that the network charge, the number of new virtual links and its assigned capacity are parameters that affect directly in the network performance and depending on these parameters the virtual topology design improve or not the network performance.
So, there is a compromise between the amount of new virtual links that is wanted to add and the wavelengths per ligthpath that minimizes the BLR, depending on the network charge.

As the simulations serve to study and validate algorithms and protocols to be applied to real traffic, it is essential to use traffic models which reflect the traffic characteristics as accurate as possible.

To obtain more realistic results with JAVOBS simulator, new traffic models has been implemented: the markov-modulated Poisson traffic model and the self-similar traffic model.

Its implementation has been motivated for the fact that although Poisson arrival processes are easy to model and analyze, they are not suitable to describe data traffic with burstyness. The Markov-modulated Poisson traffic model conveys the short-range dependence but it does not capture long-range dependence as self-similar does. So, self-similar models provide accurate and realistic description of network traffic, because it captures similar statistical properties on traffic at a range of timescales.

The developed traffic models have been validated through simulation, but it is important to continue in the research and study of the characteristics of traffic in order to optimize resource allocation strategies and obtain the best utilization of the network resources.

References

[1] Lightpath Communications: An Approach to High Bandwidth Optical WM's. I. Chlamtac, A. Ganz, and G.Karmi.

[2] Transparent Optical Packet Switching: The European ACTS KEOPS Project Approach. M Renaud, C Janz, P Gambini, C Guillemot - Proc. LEOS, 1999

[3] Optical burst switching (OBS)—a new paradigm for an optical internet. C Qiao, M Yoo - Journal of high speed networks, 1999 - IOS Press

[4] Terabit burst switching. JS Turner - Journal of High Speed Networks, 1999 - IOS Press

[5] A novel node architecture for all-optical switching networks. C Yuan, Z Li, Y He, A Xu - Proceedings of SPIE, 2007

[6] Control architecture in optical burst-switched WDM networks. Y Xiong, MLJ Vandenhoute, HC Cankaya - US Patent 6,721,315, 2004

[7] An introduction to optical burst switching. T Battestilli, H Perros - IEEE communications magazine, 2003

[8] Traffc statistics and performance evaluation in optical burst switched networks. X Yu, Y Chen, C Qiao - Proc. Opticomm, 2002 - Citeseer

[9] On Burst Assembly in Optical Burst Switching Networks- Performance evaluation analysis of OBS network with JET. K Dolzer, C Gauger - Proceedings of the 17th International Teletraffic, 2001

[10] Comparison of Conventional and Offset Time-Emulated Optical Burst Switching Architectures. M Klinkowski, D Careglio, J Solé-Pareta - … Optical Networks, 2006 International …, 2006

[11] Performance Overview of the Offset time Emulated OBS Network Architecture. M Klinkowski, D Careglio, J Solé-Pareta, M … - J. Lightwave …, 2009

[12]The Ready-to-Go Virtual Circuit Protocol: A Loss-Free Protocol for Multigigabit Networks Using FIFO Buffers

[13] Performance analysis of burst admission-control protocols. IEEE Proceeding of Communications. Widjaja.

[14] A Comparison of the JIT, JET, and Horizon Wavelength Reservation Schemes on a Single OBS Node. J Teng, GN Rouskas - Proc. 1st Int. Workshop Opt. Burst Switching, 2003

[15] JumpStart: A just-in-time signaling architecture for WDM burstswitched network. I Baldine, H Perros, GN Rouskas, D Stevenson - … , Pisa, Italy, May 19-24, 2002

[16] Just-In-Time Signaling for WDM Optical Burst Switching Networks. JY Wei, RI McFarland - Journal of Lightwave Technology, 2000

[17] Just enough time(JET): a high speed protocol for bursty traffic in optical networks. M Yoo, C Qiao - Vertical-Cavity Lasers, Technologies for a Global …, 1997

[18] A Detailed Analysis and Performance Comparison of Wavelength Reservation Schemes for Optical Burst Switched Networks. J Teng, GN Rouskas - Photonic Network Communications, 2005

[19] Control architecture in optical burst-switched WDM Networks, Xiong

[20] An overview of routing methods in optical burst switching networks

[21] A comparison study on the number of wavelength converters needed in synchronous and asynchronous all-optical switching architectures. V. Eramo, M. Listanti, and P. Pacici.

[22] Buffering in optical packet switches. D. K. Hunter, M. C. Chia, and I. Andonovic.

[23] A performance survey on deflection routing techniques for OBS network. O Pedrola, S Rumley, D Careglio, M … - Proceedings of IEEE …, 2009

[24] Offset Time-Emulated Architecture for Optical Burst Switching - Modelling and Performance Evaluation. M law Klinkowski, D Careglio, JS i Pareta, M Marciniak - 2007

[25] A performance model of deflection routing in multibuffer networks with non uniform traffic. J Bannister, F Borgonovo, L Fratta, - IEEE/ACM Transactions, 1995

[26]Performance analysis of deflection routing in OBS networks, Ching-Fang Hsu

[28] Stabilizing deflection routing in optical burst switched networks. A Zalesky, HL Vu, Z Rosberg,- IEEE Journal, 2007

[27]Performance analysis of deflection routing in optical burst switched networks. C Hsu - 2002

[29] Modelling and Performance Evaluation of Optical Burst Switched Networks with Deflection Routing and Wavelength Reservation

[30] Traffic engineering approach to path selection in obs netwotks. Jing Teng and George Rouskas, Journal of Optical Networking, Vol. 4, Issue 11, pp. 759-777 (2005)

[31] JAVOBS: A flexible Simulator for OBS Network Architectures . Oscar Pedrola, Miroslaw Klinkowski, Davide Careglio, Josep Solé-Pareta, Sébastien Rumley, ... Journal of Networks, Vol 5, No 2 (2010), 256-264, Feb 2010

[32] Traffic modeling for telecommunications networks. VS Frost, B Melamed - IEEE Communications Magazine, 1994

[33] On Multimedia Networks selfsimilar traffic and network performance. Z Sahinoglu, S Tekinay - IEEE Communications Magazine, 1999

[34] On the Self-Similar Nature of Ethernet Traffic (Extended Version). WE Leland, MS Taqqu, W Willinger, DV Wilson, M … - IEEE/ACM Transactions on …, 1994

[35] Generation self-similar traffic for opnet simulations, Arnold W. Bragg

[36] An empirical comparison of generators for self similar simulated traffic. G Horn, A Kvalbein, J Blomskøld, E Nilsen - Performance Evaluation, 2007

[37] Routing Path Optimization in OBS load balance. J Teng, GN Rouskas - Proc. 9th ONDM, 2005

[38] Book WDM optical networks (Chapter 5)

[39] A survey of virtual topology design algorithms for wavelength routed optical networks. R Dutta, GN Rouskas - Optical Networks Magazine, 2000

[40]Virtual topology design for OBS optical networks. B Wu, KL Yeung - Conference on Communications, 2007

[41] Why we should consider virtual topologies for OBS? CM Gauger, B Mukherjee - Networks, 2005 2nd International Conference on, 2005

[42] Non-linear optimization for multi-path source routing in OBS networks. M. Klinkowski, M. Pióro, D. Careglio, M. Marciniak, J. Solé-Pareta,2007

[43] Reactive and proactive routing in labeled optical burst switching networks. M KLINKOWSKI – 2009

[44]Routing Path Optimization in Optical Burst Switched Networks. J Teng, GN Rouskas - Proc. 9th ONDM, 2005

[45] Adaptive path selection in obs networks. LYGN Rouskas - J. Lightwave Technol, 2006

[46] Dynamic congestion-based load balanced routing in OBS GRV Thodime, VM Vokkarane, JP Jue - IEEE Global Telecommunications …, 2003

[47] Will Leland, Murad Taqqu, Walter Willinger, and Daniel Wilson, On the Self-Similar Nature of Ethernet Traffic (Extended Version), IEEE/ACM Transactions on Networking, Vol. 2, No. 1, pp. 1-15, February 1994.

[48] Vern Paxson, Fast, Approximate Synthesis of Fractional Gaussian Noise for Generating Self-Similar Network Traffic. Computer Communications Review, V. 27 N. 5, October 1997, pp. 5-18.

[49] Schuler, C. "fft_fgn". Research Institute for Opnet Communication Systems, GMD FOKUS, Hardenbergpatz 2, D-10623 Berlin, Germany.

[50] Vern Paxson and Sally Floyd, Wide-Area Traffic: The Failure of Poisson Modeling IEEE/ACM Transactions on Networking, Vol. 3 No. 3, pp. 226-244, June 1995.

[51] Mark E. Crovella and Azer Bestavros, Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes in IEEE/ACM Transactions on Networking, 5(6):835--846, December 1997.

[52] Performance analysis of multi-channel protocols for optical local area networks wxploting wavelength division multiplexing, Daniel Rodellar Gomez

[53] SELFIS: A Tool For Self-Similarity and Long-Range Dependence Analysis – Karagiannis, Faloutsos – 2002

[54] http://math.bu.edu/people/murad/methods/

[55] Murad S. Taqqu, Vadim Teverovsky. On estimating the intensity of long-range dependence in finite and infinite variance time series

[56] Erlang, Agner Krarup (1901). «The Theory of Probabilities and Telephone Conversations». *Nyt Tidsskrift for Matematik B* **20**.

List of Figures

Appendices

A Traffic definitions + compute network charge:

The traffic (in Erlangs) is computed as: $Traffic = traffic\ rate \cdot time\ of\ simulation$

The emitted traffic rate by each node t_n is calculated as $t_n = \rho \cdot C$

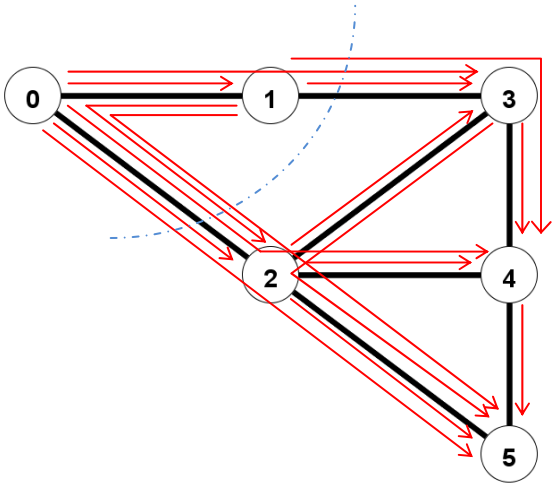
- C is the capacity of a fiber link and it is computed as $C = W \cdot R$, where W is the number of wavelengths/channels and R is the channel bitrate.
- ϱ is the traffic charge (also called traffic load), which corresponds to the occupation of the links in relation of the link capacity.

The total traffic rate in the network (T) corresponds to the traffic rate emitted for each node multiplied for the number of nodes in the network: $T = t_n \cdot N$

The emitted traffic rate on each route (τ) is calculated as: $\tau = \frac{t_n}{(N-1)} = \frac{\rho \cdot C}{(N-1)}$

Computation of different network charge in SIMPLE topology

Using the basic routing algorithm with the assumptions presented in section 2.2, the traffic flows in the SIMPLE topology are:



N is the number of nodes, N= 6;

In this graph only one port sense is represented for simplicity. For link 0↔1 there are port 0-1 and port 1-0, although only the first one is represented.

The emitted traffic rate on each route is:

$$\tau = \frac{\rho \cdot C}{(N-1)} = \frac{\rho \cdot C}{5}$$

For example, the offered traffic rate on link 0-2 corresponds to $5 \cdot 2 \cdot \tau$, where 5 are the flows (routes) in the graph which traverse the link, 2 because of the bidirectionality and τ is the individual traffic route emission.

Then, the link 0-2 is considered saturated if the offered traffic rate on the link is higher than the link capacity:

$$10 \cdot \tau \geq C \rightarrow 10 \cdot \frac{\varrho \cdot C}{(N-1)} \geq C \rightarrow \varrho \geq \frac{(N-1)}{10} \geq \frac{5}{10}$$

Thus, link 0-2 is saturated for $\varrho \geq 0.5$ and for the link 2-5 is saturated for $\varrho \geq \frac{5}{8}$

Then, for a basic routing (based on shortest path) it can be considered that:

- The network is not charged (any link is saturated) for $\varrho < \frac{5}{10} = 0.5$
- The network is low-charged when one or two links are saturated, for $\varrho < \frac{5}{8} = 0.625$
- The network is charged and high-charged for $\varrho < \frac{5}{4} = 1.25$

If a general idea is wanted to consider, independently of the routing scheme used, it is necessary to define the bottleneck of the SIMPLE topology.

The bottleneck is justified for the cut (in blue) which partitions the network in {0,1} and {2,3,4,5}. In this case, through 2 links (1-3 and 0-2) have to travel the routes from 0 and 1 to the rest of nodes, so:

$$(2\text{sources} \cdot 4\ \text{destinations}) \cdot 2(\text{bidirecc}) \cdot \tau = 2 \cdot C$$

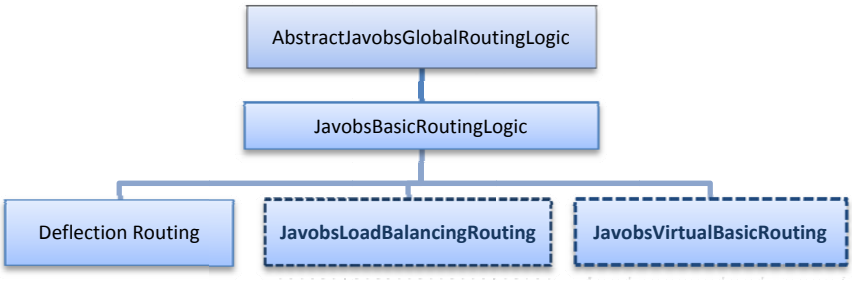
$$16 \cdot \frac{\varrho \cdot C}{(N-1)} = 2 \cdot C \quad \rightarrow \quad \rho = \frac{N-1}{C} = \frac{5}{8} = 0.625$$

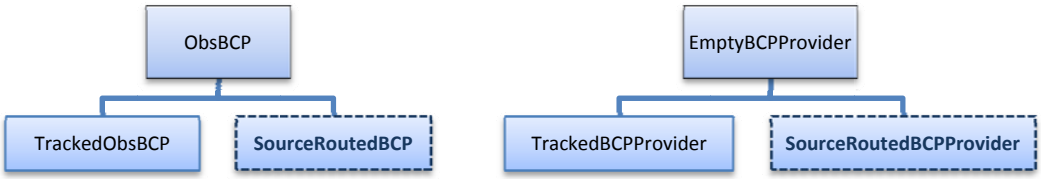
In conclusion, for any routing, the network is considered

- low-charged for $\varrho < 0.625$
- charged for $\varrho > 0.625$

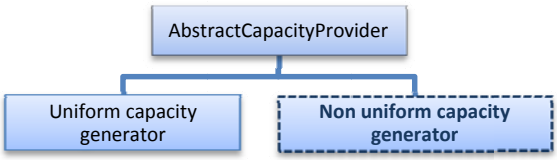
B Javobs hierarchy of the new implemented class

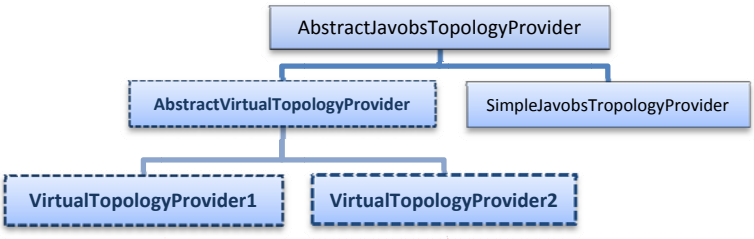
ROUTING principal hierarchies:



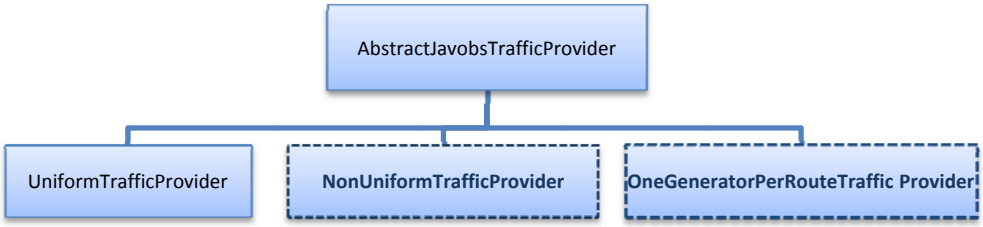


VIRTUAL TOPOLOGGY principal hierarchies:





TRAFFIC principal hierarchies:



MARKOV

SELF-SIMILAR