UNIVERSITAT POLITÈCNICA DE CATALUNYA

A Prediction-Based Routing Mechanism for Optical and IP/MPLS Networks

Eva Marín Tordera

June 2007

UNIVERSITAT POLITÈCNICA DE CATALUNYA

A Prediction-Based Routing Mechanism for Optical and IP/MPLS Networks

Departament d'Arquitectura de Computadors

Ph.D. Student: Eva Marín Tordera

Advisors: Xavier Masip Bruin,
Sergio Sánchez López

A THESIS PRESENTED TO THE UNIVERSITAT POLITÈCNICA DE CATALUNYA
IN FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
Doctora per la UPC

June 2007

ACTA DE QUALIFICACIÓ DE LA TESI DOCTORAL

Reunit el tribunal integrat pels sota signants per jutjar la tesi doctoral:

Títol de la tesi: A Prediction-Based Routing Mechanism for Optical and IP/MPLS Networks

Autor de la tesi: Eva Marín Tordera ....

Acorda atorgar la qualificació de:

- □ No apte
- □ Aprovat
- □ Notable
- □ Excel·lent
- □ Excel·lent Cum Laude

Barcelona, …………… de/d'…..................……………. de ....…

El President	El Secretari
....	....
(nom i cognoms)	(nom i cognoms)

El vocal	El vocal	El vocal
....	....	....
(nom i cognoms)	(nom i cognoms)	(nom i cognoms)

Acknowledgements/ Agradecimientos

Quisiera comenzar los agradecimientos por mi hija Violeta, porque ella en cierta medida es el origen de esta Tesis. A parte de agradecerte ser el motor de mi vida que impulsa y alienta todo lo que hago, además sin ti quizá nunca me hubiese metido en el mundo de las redes. Fue a partir de tu nacimiento que me replanteé el doctorado y cambié mi tema de Tesis.

Naturalmente también tengo que agradecer a mis directores de Tesis, Sergio y Xavi que me hayan alentado y aceptado en su grupo de investigación. Además tengo que agradecerles la libertad que me han dado a la hora de incorporar las ideas de Arquitectura de Computadores en el mundo de las redes; así como la guía y ayuda que me han ofrecido.

Quiero agradecer también a todos los miembros del grupo de Comunicaciones en Banda Ancha del Departamento de Arquitectura de Computadores y de otros departamentos, especialmente a Josep Solé y Jordi Domingo directores de la línea, por la buena acogida que siempre me han dado y por los consejos que he recibido.

Me gustaría también recordar a mis antiguos directores de Tesis, Juanjo y Toni, por su apoyo y por haberme enseñado a investigar.

También quiero agradecer a todos mis compañeros de la delegación de Arquitectura de Computadores en Vilanova i la Geltrú.

Por último quiero agradecer a mi familia, especialmente a mi madre, por todo su apoyo y ayuda.

Resumen

En los últimos años, las nuevas aplicaciones de Internet tales como aplicaciones multimedia, video a demanda, etc, requieren progresivamente mayor capacidad y garantías de calidad de servicio. En ese sentido el modelo de transmisión de tráfico ha ido cambiando hacia Redes Ópticas las cuales proporcionan mayor capacidad y fiabilidad.

Esta Tesis se encamina a proporcionar nuevos mecanismos de routing basados en conceptos predictivos para ser aplicados tanto en Redes Ópticas como redes *IP/MPLS*.

El proceso de routing implica seleccionar la ruta (o ruta y longitud de onda en Redes Ópticas) que mejor transporta la información desde el nodo fuente hasta el nodo destino en una red. El routing en redes *IP/MPLS* se conoce como *QoS* (Quality of Service) routing cuando calcula rutas que requieren ciertas garantías de calidad de servicio. Por otro lado, el routing en redes ópticas debe seleccionar no sólo el camino físico o ruta sino también la longitud de onda por donde el tráfico debe de ser transportado (conocido como problema de Routing and Wavelength Assignment, *RWA*). Con el propósito de introducir el escenario del problema ilustraremos el caso para redes ópticas.

Las más recientes soluciones propuestas en la literatura para el problema de Routing and Wavelength Assignment, *RWA*, utilizan mecanismos distribuidos basados en routing de fuente. En routing distribuido, los nodos fuentes seleccionan la ruta y la longitud de onda basándose en la información de estado de la red contenida en sus bases de datos. En este escenario aparece el problema del routing inexacto (routing inaccuracy problem) porque por diferentes razones esta información de estado de la red contenida en las bases de datos no es exacta. El problema del routing inexacto describe el impacto en el rendimiento global debido a tomar decisiones *RWA* a partir de información inexacta o desactualizada. En general, una parte importante de un mecanismo de routing es la política de actualización. En routing distribuido los nodos fuentes deben intercambiar información sobre los recursos (ancho de banda disponible o longitudes de onda disponibles) de sus enlaces. En la literatura hay diferentes propuestas tratando el problema del routing inexacto. Estos trabajos proponen tanto nuevos algoritmos de routing como nuevas políticas de actualización. Acerca de las políticas de actualización, cuando la frecuencia de actualización de las bases de datos es alta, la información de estado de la red será más

precisa. Pero debe existir un compromiso entre la frecuencia de actualización y la sobrecarga de señalización generada por los mensajes de actualización en la red. Incluso, asumiendo en una red óptica que la sobrecarga de señalización no es un problema porque una fibra o longitud de onda se dedica a tareas de señalización (fuera de banda), es posible que la información no sea completamente precisa. Existe un tiempo mínimo de propagación necesario para diseminar esta información en la red y para que se estabilice esta información en las bases de datos.

Por otro lado, hasta ahora Internet sólo proporcionaba un modelo de transmisión 'best effort'. Las aplicaciones a tiempo real mencionadas no pueden ser soportadas en este modelo 'best effort', ya que requieren cierto grado de calidad de servicio. El volumen de información que los nodos fuente deben intercambiar cuando se tienen en cuenta los parámetros de calidad de servicio es mayor, y esto impacta negativamente en la sobrecarga de señalización.

Esta Tesis propone un nuevo mecanismo de routing, llamado Prediciton-Based Routing (*PBR*), basado en conceptos predictivos, el cual no necesita mensajes de actualización con información de estado de la red. La principal idea subyacente es que a frecuencias de actualización asequibles la información de estado obtenida en los mensajes de actualización puede no ser útil. Por lo tanto, no utilizar esta información es mejor ya que su efecto tiene un impacto negativo en el rendimiento global de la red.

El mecanismo Prediction-Based Routing (*PBR*) propuesto en esta Tesis tiene el propósito de reducir tanto la sobrecarga de señalización como los efectos negativos del problema del routing inexacto. La información de estado utilizada por los nodos fuentes no se actualiza mediante mensajes sino que es deducida del comportamiento de peticiones de conexión previas. Es decir, el mecanismo *PBR* tiene en cuenta los bloqueos de conexión previamente producidos en el mismo 'lightpath' (ruta y longitud de onda). Además, otra importante característica del *PBR* es su simplicidad comparado con algoritmos propuestos previamente.

Index

PART I. INTRODUCTION

PART II. WDM NETWORKS

PART III. IP/MPLS NETWORKS

PART IV. CONCLUSIONS AND FUTURE WORK

List of Abbreviations

ASON	Automatic Switched Optical Network
ATM	Asynchronous Transfer Mode
BAPHOR	Balanced Predictive Hierarchical Optical Routing
BBOR	BYPASS Balanced Optical Routing
BHOR	Balanced Hierarchical Optical Routing
DiffServ	Differentiated Services
DoS	Degree of Service
ESCON	Enterprise Systems Connection
FDL	Fibre Delay Line
FF	First Fit
IETF	Internet Engineering Task Force
IP	Internet Protocol
IntServ	Integrated Services
ITU-T	International Telecommunication Union-Telecommunication Sector
LAN	Local Area Network
LER	Label Edge Router
LLR	Least Loaded Routing
LRA	Logical Routing Area
LSP	Label Switched Path
LSR	Label Switched Router
MPLS	Multiprotocol Label Switching
MTE	Multilayer Traffic Engineering
NAS	Node Aggregation Scheme
OADM	Optical Add and Drop Multiplexer
OIF	Optical Internet Working Forum
OSPF	Open Shortest Path First
OTL	Optical Line Terminal
OTN	Optical Transport Network
OXC	Optical Cross Connect

PBR	Prediction-Based Routing
PC	Program Counter
PHOR	Predictive Hierarchical Optical Routing
POW	Potentially Obstructed Wavelength
PSR	Predictive Selection of Route
PT	Prediction Table
QoS	Quality of Service
RA	Routing Area
RAL	Routing Area Leader
RFC	Request For Comments
RSVP	Resource Reservation Protocol
RWA	Routing and Wavelength Assignment
RWP	Route and Wavelength Prediction
SLE	Static Lightpath Establishment
SONET	Synchronous Optical Network
SDH	Synchronous Digital Hierarchy
SP	Shortest Path
TCP	Transmission Control Protocol
TE	Traffic Engineering
WAN	Wide Area Network
WDM	Wavelength Division Multiplexing
WI	Wavelength Interchangeable
WS	Wavelength Selective
WR	Wavelength Register
WSP	Widest Shortest Path

List of Figures

List of Tables

Abstract

In the last years, new Internet applications such as multimedia, video on demand, multimedia conferences, triple play, gaming and virtual reality increasingly request greater capacity and guarantees of traffic delivery in such a way that the traffic transmission model is moving towards Optical Networks which provide high capacity and reliability.

This Thesis aims at providing a new routing mechanisms based on prediction concepts to be applied to both Optical and *IP/MPLS* networks.

The routing process implies to select the route (or route and wavelength in Optical Networks) that can best transport the information from the source node to the destination in a network. Routing in IP/MPLS networks is known as *QoS* (Quality of Service) routing when computing routes for clients requiring traffic delivery guarantees. On the other hand the routing process in Optical Networks implies to select not only the physical path but also the wavelength where the traffic will be transported, known as the Routing and Wavelength Assignment (*RWA*) problem. In order to introduce the problem we will illustrate the case for Optical Networks.

Routing and Wavelength (*RWA*) solutions recently proposed in the literature use distributed mechanism based on source routing. In distributed source routing, source nodes select the route and the wavelength based on the network state information contained in their network state database. In this scenario the routing inaccuracy problem comes up because for different reasons this network state information is not accurate. The routing inaccuracy problem describes the impact on global network performance because of taking *RWA* decisions according to inaccurate or outdated information. In distributed source routing source nodes must exchange periodically information about the resources (available bandwidth or available wavelengths) on their links. In general an important part of a routing mechanism is the update policy. In the literature there are different proposals dealing with the routing inaccuracy problem. These works propose both, new routing algorithms and new update policies. Concerning to the update policy, when the frequency of updating the network state databases is high the network state information contained is more accurate. But it may exist a trade-off between the frequency of updating and the signalling overhead produced by these update messages in the network. Even, assuming in

an optical network that the signalling overhead is not a problem because a fibre or a wavelength is dedicated to signalling tasks (out-of-band), it is possible that the information is not completely accurate. It exists a minimum propagation time needed to disseminate and to stabilize this information in the network databases.

On the other hand, till now the Internet only provides a best effort transmission model. The mentioned real time applications can not be supported by this best effort model requiring a certain degree of Quality of Service (*QoS*). The volume of information that source nodes exchange when the *QoS* parameters are included is larger, impacting negatively on the signalling overhead.

This Thesis proposes a new routing mechanism, named the Prediction-Based Routing (*PBR*) based on prediction concepts, which does not need network state update messages. The main underlying idea is that at affordable update frequencies the network state information obtained from the update messages can not be so useful. Hence, we end up showing that not using this information is better because of its negative effects on the network performance.

The Prediction Based Routing (*PBR*) mechanism is aimed to reduce both, the signalling overhead and the negative effects of the routing inaccuracy problem. The network state information managed by the source nodes is not update by means of update messages but it is inferred from the behaviour of the previous connection requests. The *PBR* mechanism takes into account the previous blocked connections produced in the same lightpath (route and wavelength) in optical networks and route in *IP/MPLS* networks. Another important characteristic of the *PBR* is its simplicity compared with previous proposed algorithms.

PART I

INTRODUCTION

In order to place this Thesis in context, this part summarizes the networks evolution and reviews the concepts of Traffic Engineering, Quality of Service, etc. Moreover, it contains a brief introduction of branch prediction in computer architecture. Finally, it presents the main objectives of this Thesis and its organization.

1. Networks Evolution

In information technology, a network is a series of points or nodes interconnected by communication paths. Networks can interconnect with other networks and contain subnetworks. In general, networks can be divided into two types: connection-less oriented networks and connection oriented networks. A clear example of the first type of network is Internet, being the telephony network a currently active example of the second one. In a connection-less oriented network neither circuit nor path should be established in order to send data. These networks are also known as packet switched networks. Instead, in connection oriented networks, also known as circuit switched networks, it is necessary to establish a connection or circuit before sending any data.

Internet is the worldwide, publicly accessible network of interconnected computer networks that transmit data by packet switching using the standard Internet Protocol (*IP*). It is a network of networks supporting different types of subnetworks and protocols. Just as a definition, a computer 'is in Internet' if it executes the set of *TCP/IP* protocols, has an *IP* address and can send *IP* packets towards all the other computers in Internet [1].

On the other hand, Internet is divided into autonomous systems (*AS*). An autonomous system (*AS*) is a collection of *IP* networks and routers under the control of one entity that presents a common routing policy. The term routing refers to selecting those routes or paths used to forward data. The division into *ASs* gives to Internet a hierarchical structure. For technical, managerial, and sometimes political reasons, the Internet routing system consists of two components, interior routing and exterior routing. The concept of an Autonomous System (*AS*) plays a key role in separating interior from exterior routing. Interior gateway protocols (*IGPs*) are used to distribute routing information within an *AS* (i.e., intra-*AS* routing or intra-domain). Exterior gateway protocols are used to exchange routing information among *ASs* (i.e., inter-AS routing or inter-domain). This Thesis will focus on Interior Routing Protocols.

Internet was thought up in a military environment and its main characteristic was to be tolerant to failures. Data is divided and then sent into packets of information. Each packet is independently switched and can follow a different path to reach the destination. In the case that some router fails the network can continue working. One of the main advantages of a packet switched network is that a router failure does not motivate the network to crash.

However, unlike packet switched networks, in a circuit switched network if a router fails the connection can be lost. Despite this weakness circuit switched networks has some advantages to transport data. The main advantage focuses on its facility to provide the network with Quality of Service features. The term Quality of Service (*QoS*) refers to the set of service requirements to be met by the network while transporting a connection or flow. Resource reservation indeed can be simultaneously done when establishing the connection. Examples of circuit switched networks supporting data transport are X.25, Frame Relay and *ATM* (Asynchronous Transfer Mode) [3]. One might say that *ATM* was actually the first technology offering Quality of Service capabilities.

Nowadays Internet is completely extended around the world despite Quality of Service is not one of its features and hence only best effort service is provided. During many years several initiatives have been proposed to provide the current Internet model with Quality of Service capabilities. Some of these initiatives take up again the idea of connection oriented networks due to its facility for resource reservation. It is worth highlighting the network model arising from applying the Multiprotocol Label Switching (MPLS) [4], known as *IP/MPLS*. As done in a circuit switched network, in *IP/MPLS* networks the connection has to be established before sending the information which is switched in terms of packets all following the same previously established path.

Besides, the development of the optical fibre involved firstly the improvement of physical layer in terms of transmission rates. *SONET/SDH* [5][6] came up to deal with the high bandwidth supported by the optical fibre as well as to provide network flexibility. Hence, *SONET/SDH* networks were designed as the first generation of optical networks. Today, *SONET/SDH* is strongly implemented as the core of the telecommunications in North America and Europe [2].

The research done in order to incorporate some of the switching and routing functions from the electronic domain to the optical domain has allowed Wavelength Switched Optical Networks to be developed. Similar to circuit switched networks an optical connection has to be established before data can be transmitted. An optical connection consists on a wavelength on a route or path, also known as lightpath.

Finally, Optical Burst Switched Networks and Optical Packet Switched Networks were thought up in order to incorporate both the advantages of optical networks and packet switched networks.

1.1. Optical Networks

An optical network is a communication network in which data is transmitted over fibre optic lines as pulses of light. The optical network provides high capacities needed for new applications such as those coming from the residential users (multiplay, gaming, VoD), from the business users (ubiquitous application services) and those not yet defined but clearly foreseen by most operators and providers. Optical networks achieve this high capacity by means of the multiplexing technique called wavelength-division multiplexing (*WDM*). The idea of wavelength-division multiplexing (*WDM*) is to transmit data simultaneously at multiple carrier wavelengths (or colours) over a single fibre. *WDM* provides ‘virtual circuits’, and a single fibre looks like multiple ‘virtual circuits’ each one carrying a different stream of data. In this sense with *WDM* it is possible to transmit data at higher rates over a single fibre.

The optical networks can be divided into two generations. The former uses the optical fibre as a replacement of copper cable to get higher capacities. The optical fibre provides much higher bandwidth and lower bit error. Examples of this first generation of optical networks are *SONET/SDH* networks. *SONET/SDH* networks indeed simply use the lightpath (wavelength and route) provided by the optical network as a replacement of the usually fixed fibre connections between *SONET/SDH* terminals. In every node, *SONET* terminal, it is necessary an optoelectronic conversion to process the data, which is processed in the electronic format and then back again to optical signal. This lack of optical processing capabilities results in reducing the processing speed and the scalability. Most of the firstly developed long distance *IP* architectures were based on either *SONET/SDH* or *ATM* over *SONET/SDH*. The *IP* packets or the *ATM* cells carrying *IP* packets were encapsulated in *SONET/SDH* frames

The latter provides circuit-switched lightpaths by routing and switching wavelengths inside the network [2]. A wavelength routed *WDM* (Wavelength Division Multiplexing) network is a circuit-switched network, in which a lightpath (wavelength and route) must be established between a source-destination node pair before data can be transferred. The idea of wavelength-division multiplexing (*WDM*) is to transmit data simultaneously at multiple carrier wavelengths over a single fibre. A lightpath is an end-to-end optical connection between a source-destination node pair, which may span multiple fibre links and use a single or multiple wavelengths. An Optical Transport Network (*OTN*) consists of switching nodes (Optical Cross-Connect, *OXC*) interconnected by wavelength-division multiplexed (*WDM*) fibre-optic links that provide multiple huge bandwidth communication channels over the same fibre in parallel. *OXCs* are able to switch wavelengths from one input port to another of their large number of ports. Other optical network elements being part of an *OTN* are the Optical Line Terminals (*OLTs*) and optical add/drop multiplexers, *OADM*. An *OLT* multiplexes multiple wavelengths into a single fibre, and demultiplexes a set of wavelengths of a single fibre into separate fibres. An *OADM* has two line ports and selectively drops some of the wavelengths of the input port and also adds new wavelengths to composite a *WDM* signal to the output port. The optical networks provide lightpaths to its users, such as *SONET* terminals, *IP* routers or *ATM* switches. When the *OTN* includes automatic switching capabilities, it is referred to as an Automatically Switched Optical Network (*ASON*). This *ASON* capability is accomplished by using a control plane that dynamically set up or tear down the optical connections. G/MPLS [7] is the protocol included in the *ASON* recommendation [27].

One of the objectives of this second generation of optical networks is to reduce most of the intermediate layers and to map directly the *IP* packets over optical lightpaths. Moreover the new *WDM* networks will have to support other network protocols such as *IP/MPLS* [4], *ATM* [3], *SONET/SDH* [5][6], Gigabit Ethernet [8], *ESCON* (Enterprise Systems Connection of IBM) [9], etc, all coexisting on the same fibre. See Figure 1 as an example. In a classical layered view, the first generation of optical networks provided only those functions corresponding to the physical layer, but this second generation of optical network provides services that correspond to the link and the network layer. These services include the yet mentioned, switching and routing capabilities and also monitoring and fault recovery facilities. In that case where data information can be transmitted by a lightpath from the source node to the destination node without needing optoelectronic conversion in any point of the path, this optical network is referred as all-optical network. Otherwise, when optoelectronic conversion is required in every node of the path, this network is named as opaque. This is the case of the first generation of optical networks such as *SONET/SDH*. Nowadays the optical networks are semitransparent networks, composed by some all-optical subnetworks and some opaque subnetworks.

1.2. *IP/MPLS*

In the traditional *IP* network layer, the header of each packet is analyzed, and the next node (hop) is chosen based on a routing table. Multiprotocol Label Switching (*MPLS*) [4] provides a mechanism that is independent of routing tables. *MPLS* assigns short labels to network packets that describe how to forward them through the network. In an *MPLS*

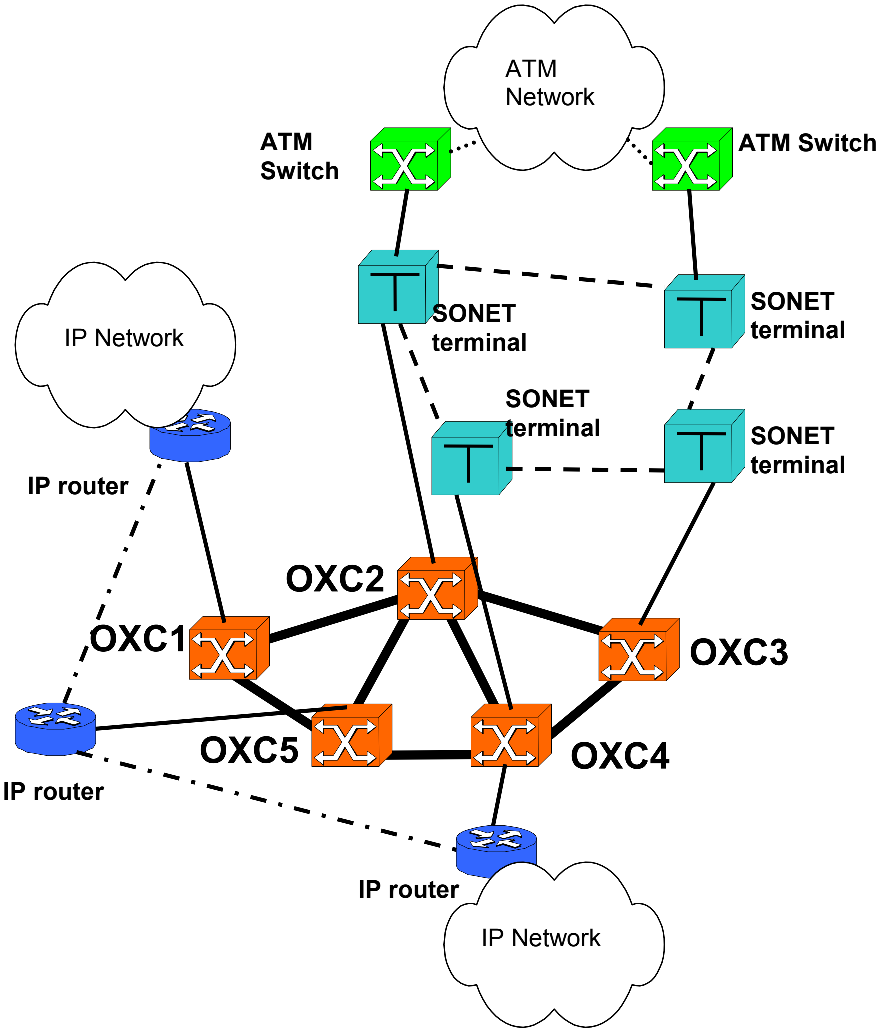


Figure 1. Example of Optical Network. *IP* routers, and *SONET* terminals request lightpaths to the Optical Network.

environment, the analysis of the packet header is performed just once, when a packet enters in the *MPLS* network. Then, the packet is assigned to a stream, which is identified by a label, which is a short (20-bit), fixed length value at the front of the packet. Labels are used as lookup indexes into the label forwarding table. For each label, this table stores forwarding information. Hence, *MPLS* decouples the routing and forwarding functionality.

MPLS provides virtual circuits to support end-to-end traffic streams. A virtual circuit forces all the packets belonging to that circuit to follow the same path through the network, allowing better allocation of resources in the network. Unlike a real circuit-switched network, a virtual circuit does not provide fixed guaranteed bandwidth along the path of the circuit due to the fact that statistical multiplexing is used to multiplex virtual circuits inside the network.

One of the main advantages of *MPLS* is that the routing process is significantly simplified. *MPLS* was designed to work directly on *ATM* switches or *IP* routers. *MPLS* has functionalities comparable to *ATM QoS* (Quality of Service) capabilities, becoming the *IP* networks more than a best effort network.

2. *TE* and *QoS* Routing

As mentioned in last Section, current Internet only supports a best effort transmission model. In best-effort service the network tries its best to send the data from the source to the destination as quickly as possible, but offering no guarantees. This best-effort service is the most usual in Internet today and it is useful for different applications such as web browsing or file transfer. However, the best-effort service is not adequate for highly delay sensitive applications, such as real time video, multimedia, voice calls, multimedia conferences, triple play, gaming and virtual reality.

These new network applications have requirements in terms of delay, congestion, blocking, packet losses, etc that cannot be supported by the current network model. Traffic Engineering aims to optimize the performance of networks by improving the utilization of network resources. According to the *RFC* 2702 [10] Traffic engineering is defined as:

"Traffic Engineering (TE) is concerned with performance optimization of operational networks. In general it encompasses the application of technology and scientific principles to the measurement modelling, characterization, and control of internet traffic, and the application of such knowledge and techniques to achieve specific performance objectives. A major goal of Internet Traffic Engineering is to facilitate efficient and reliable network operations while simultaneously optimizing network resource utilization and traffic performance. Traffic Engineering has become an indispensable function in many large autonomous systems because of the high cost of networks assets and the commercial and competitive nature of Internet. These factors emphasize the need for maximal operational efficiency"

Traffic Engineering controls the network's response to traffic demands and other stimuli, such as link or node failures. One of the main*TE* functionalities is to provide Quality of Service. Yet defined in first section, the Quality of Service (*QoS*) is a set of service requirements to be met by the network while transporting a connection or flow, but it can also be defined as the collective effect which determines the degree of satisfaction of a user of the service.

According to the *RFC* 2216 of 1997 [11], the Quality of Service, *QoS*, is formally defined as follows:

"Quality of service refers to the nature of packet delivery service provided, as described by parameters such as achieved bandwidth, packet delay, and packet loss rates. Traditionally, the Internet has offered a single quality of service, best-effort delivery, with available bandwidth and delay characteristics dependent on instantaneous load. Control over the quality of service seen by applications is exercised by adequate provisioning of the network infrastructure. In contrast, a network with dynamically controllable quality of service allows individual application sessions to request network packet delivery characteristics according to their perceived needs, and may provide different quality of service to different applications. It should be understood that there is rage of useful possibilities between the two endpoints of providing no dynamic QoS control at all and providing extremely precise and accurate control of QoS parameters".

One example of packet switched network with *QoS* capabilities is *ATM*. *ATM* provides a connection oriented service (virtual circuits) capable of providing a variety of quality of service guarantees

Over the past decade, a significant amount of work has been dedicated to provide *QoS* in *IP* networks. Examples of this work are the proposals of Integrated Services (*Intserv*) [12] and Differentiated Services (*Diffserv*) [13] architectures both by the Internet Engineering Task Force (*IETF*). The *Intserv* model achieves the *QoS* guarantees through end-to-end resource reservation by performing per-flow scheduling in all intermediate nodes. The Resource Reservation Protocol (*RSVP*) [14] is an *Intserv* signalling protocol used by both the end clients to demand their *QoS* needs according to the defined *Intserv* service classes and the core network to handle the path establishment. On the other hand, the *Diffserv* model [13] proposed by the *IETF* defines a number of per-hop behaviours that enable providing relative *QoS* guarantees for different classes of traffic aggregates.

The usual *QoS* mechanisms proposed for *IP* networks are not easily applied to *WDM* networks mainly due to the fact that these approaches are based on the store-and-forward model and mandate the use of buffers for contention resolution. Currently there is not yet optical memory and the use of electronic memory in an optical switch needs optical-to-electrical (O/E) and electrical to optical (E/O) conversions within the switch. In fact, despite *FDLs* (Fibre Delay Lines) can support a limited buffering capacity, *FDLs* do not have enough buffering capability to cope with the required *QoS* approaches. There are different proposals for *QoS* provisioning in *WDM* networks. These mechanisms take into account the physical characteristics and limitations of the optical domain. A review of some of the first proposals can be found in [15]. An example of them is the Differentiated Optical Service (*DoS*) model. More recent proposals are those presented in [16], [17] and [18].

When routing includes QoS features is known as QoS routing. *QoS* routing allows the network to determine a route that supports the *QoS* requirements of one or more flows in the network. A flow can be either a flow of *IP* packets, or an *MPLS* connection, or an optical connection, etc. The current internet intra-domain protocols such as *OSPF* [19] selects the "shortest route", basically in terms of number of hops, without taking into account the resource availability or any other required constraints. This means that flows can be routed over paths that are unable to support the flow requirements (blocking of the connection request), while other paths with available resources are not selected.

This Thesis will be focussed on routing mechanisms, *RWA* mechanisms for Optical Networks and *QoS* routing mechanism (bandwidth constraint) for *IP/MPLS* networks.

In Optical Networks the objective of a *RWA* algorithm is to select the more feasible lightpath (route and wavelength) with more probability of reaching the destination without blocking of the lightpath request. On the other hand the objective of *QoS* routing algorithms is to find paths that satisfy a given set of *QoS* constraints, such as bandwidth, delay, delay jitter or packet loss probability and also minimizing the blocking probability. The problem of finding a route with multiple *QoS* constraints is known to be NP-complete. However in the literature it can be found a lot of proposed *QoS* routing algorithms supporting multiple constraints based on heuristics. The main problem of these algorithms is their complexity.

In general a usual routing mechanism is divided into two tasks: 1) Collect the network state information and keep it updated, 2) Compute the feasible path for every new connection request. According to where the routing algorithm computes the paths and where the network state information is kept there are two categories of routing, centralized routing and distributed routing. In centralized routing the routes are computed in a single node which keeps all the network state information. Instead, in distributed routing all the nodes of the network can compute routes and keep network state information. Besides, routing can be classified in explicit (source/destination) or hop-by-hop routing. While all intermediate nodes are defined in the former only the next hop is defined in the latter. This

Thesis is focused on distributed source routing. In this case, the routes are computed in the source nodes, being completely defined from source to destination. Moreover, every source node maintains network state information based on which the routing algorithm computes the routes.

This network state information maintained in the source nodes can be a global image of the network state. That is, information about the availability on all the links of the network topology. But, this network state information can be only partial information about the links of the network, or even only local information. Local information means network state information from the point of view of such a source node. When nodes' databases maintain partial or global network state information about the links of the network, this information has to be exchanged by flooding update messages between among network nodes. These update messages contain information about the changes (new bandwidth allocated, new delay, new number of free wavelengths, etc) produced in the links of each node.

The main disadvantages of distributed source routing are on the one hand, the high signalling overhead produced by the network state information updating process; and on the other hand the inaccuracy of this network state information. The routing inaccuracy problem describes the impact on global network performance because of taking routing decisions according to inaccurate information. This inaccuracy is mainly produced by having outdated network state information (delay of propagation and triggering of the update messages) and by the aggregation process performed before flooding the information.

3. Basics of Branch Prediction

One of the organizational approaches in computer systems to achieve greater performance is the instruction pipelining. Instruction pipelining is similar to the use of an assembly line in manufacturing plant. The instructions are divided into a number of stages which occur in sequence. The various stages will be more nearly equal duration. Different instructions can be executed in different stages of the pipelining. Assuming instructions follow an implicit sequence, each cycle an instruction ingresses in the pipeline, and after a transitory time each cycle an instruction will egress the pipelining, achieving the rate of execution of one instruction per cycle. See Figure 2 as an example of pipeline of six stages. However this rate of execution is unlikely for different reasons such as the conditional branch instructions. A conditional branch instruction breaks the implicit sequence. It computes the address of the next instruction to be fetched and also checks any condition to know which this next instruction is. The fetch stage must wait until receiving the next instruction address from a more advanced stage. In the example of Figure 2 it is assumed that instruction address and condition are computed in the third stage of the pipeline, losing two cycles in every conditional branch instruction. This lost time can be reduced by guessing. A simple rule is the following. When a conditional branch is passed from the fetch stage to the next, the fetch stage fetches the next instruction in sequence. Then, if the branch is not taken, no time is lost. If the branch is taken, the fetched instructions must be discarded and a new instruction is fetched. [20].

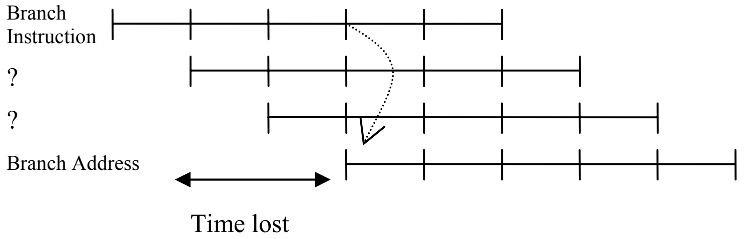


Figure 2. Example of pipelining of 6 stages.

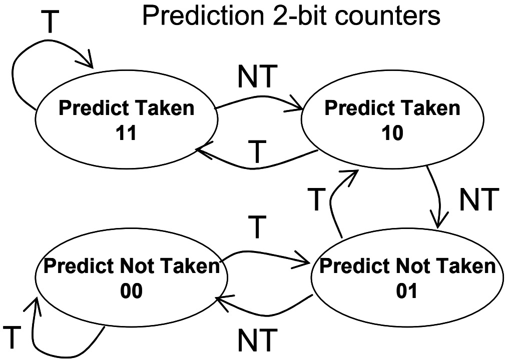
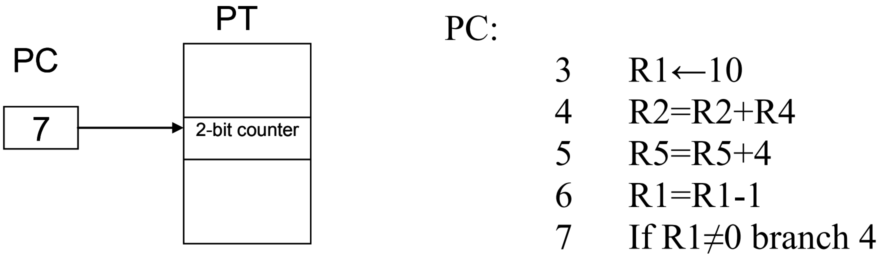


Figure 3. Branch Prediction, state diagram using two-bit counters.

A variety of approaches have been taken for dealing with conditional branch instructions; one of these techniques is branch prediction. Branch prediction deals with predicting whether a branch will be taken or not, that is the outcome of the branch. (It is worth mentioning that in the following description it is not considered the prediction of instruction address, only the outcome). These techniques include static prediction, always the branch instruction is predicted taken (or not taken), or dynamic prediction. Dynamic branch prediction strategies attempt to improve the accuracy of prediction by taking into account the previous history of conditional branch instructions in a program. The easier example is to associate 1 or more bits in a Prediction Table, *PT*, with each conditional branch instruction (identified by the address of the instruction, that is the Program Counter, *PC*). These bits reflect the recent history of the branch instruction and are referred to as a taken/not taken switch that directs the processor to make a particular decision when the same branch instruction is encountered. With a single bit, all that can be recorded is whether the last execution of this instruction resulted in a branch taken or not. If two-bits are used (See Figure 3), they can be used to record a state representing the result of the last two instances in the execution of the associated branch. In Figure 3 we can see the finite state machine when using two bits. If the past two times the given branch instruction takes the same path, taken or not taken, the prediction is to take again the same path. If the prediction is wrong, it remains the same the next time the branch instruction is encountered. If the prediction is wrong again, the prediction will be to select the opposite path. Thus, the algorithm requires two consecutive wrong predictions to change the prediction decision.



```
PC:
     3    R1←10
     4    R2=R2+R4
     5    R5=R5+4
     6    R1=R1-1
     7    If R1≠0 branch 4
```

Figure 4. Example of loop finishing in a branch instruction.

Figure 4 represents an example of loop which finishes with a conditional branch instruction, instruction of *PC*=7. Assuming that initially all the two-bit counters are 00, the first time the processor executes the branch instruction the prediction is ‘not taken’, but the branch is taken. The finite state machine changes to the state 01. The second time the loop is executed the prediction is ‘not taken’ again. The branch is really taken and the finite state machine changes to the state 10. The third time the loop is executed the branch is predicted ‘taken’ and the prediction is correct. The finite state machine is updated to state 11. From the fourth to the ninth times that the loop is executed, the prediction is correct and the branch is taken. The tenth time the processor executes the branch instruction finishing the loop the prediction is ‘taken’ but the branch is really not taken. Then, the finite state machine changes its state to 10. In this example of loop the branch is correctly predicted seven times of the ten of the loop. Assuming that there is not penalty in time when a prediction fails, and also assuming 2 cycles lost per branch without prediction, the processor is saving 14 cycles by means of the branch predictor.

In the above explanation the index used to access the Prediction Table and then the two-bit counter is built from either all or some of the bits of the Program Counter, *PC*, of the branch instruction. Notice that the *PC* is the memory address of the instruction. This means that the branch instruction and then its two-bit counter are associated with its memory address. This is not the only option proposed to access the two-bit counters. Another different possibility is to keep in a history register the past behaviour of the branch. This behaviour is registered in vectors that hold 0s and 1s, 0 if the branch is not taken and 1 if it is taken. Usually there is one of these registers for every different branch instruction, this is known as local history. These history registers are used to both, access the Prediction Tables and to update these Prediction Tables. Figure 5 represents an example of accessing to the Prediction Tables (*PTs*) by means of the Branch History Registers (BH). It is

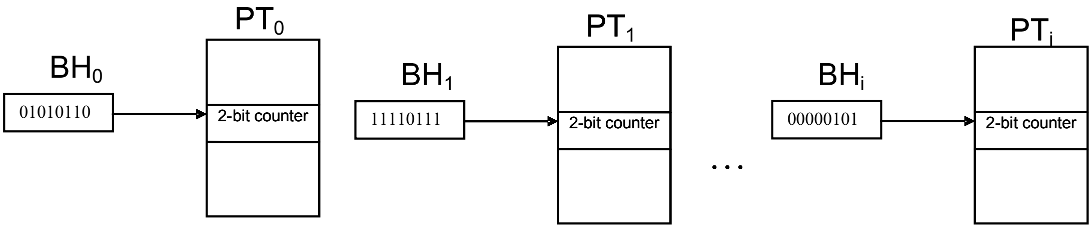


Figure 5. Branch Prediction using Branch History Registers.

assumed that there is one of such a BH registers and a *PT* for every different branch instruction. Every Prediction Table has different entries each one corresponding to a different pattern history. The main idea is that if the history is repeated the outcome of the branch can be predicted. When a branch instruction is encountered, its Prediction Table is accessed and read by means of the index obtained from its corresponding Branch History Register (BH). The prediction is done based on the read value. When finally the branch instruction is resolved and the branch direction is known the Prediction Table is updated.

4. Objectives and Organization of this Thesis

This Thesis focuses on proposing, describing, validating and verifying a routing mechanism based on prediction concepts, the Prediction-Based Routing (*PBR*) mechanism, that aims to minimize the amount of signalling messages while reducing the effects of routing under inaccurate routing information. The mechanism is applied to both Optical and *IP/MPLS* networks.

The network state information managed by usual routing algorithms is not so accurate for different reasons. The routing inaccuracy problem describes the impact on global network performance because of taking routing decisions according to inaccurate or outdated information. In this Thesis it is argued that in distributed source routing and highly dynamic traffic the network state information might never be completely accurate.

The novel idea of the *PBR* mechanism is the fact that it brings the branch prediction concepts used in computer architecture to a network scenario. Note that, in branch prediction the future behaviour of the branch instructions can be inferred from the previous behaviour. The *PBR* mechanism is aimed to reduce both, the signalling overhead due its independence from update messages, and the negative effects of the routing inaccuracy problem.

The Prediction-Based Routing (*PBR*), without update messages and with low complexity, outperforms usual routing mechanisms in different network topologies, traffic loads and resources availability.

The initial idea to bring the branch prediction concepts to a networks scenario, was to modify the branch prediction scheme that uses branch history registers (*BH* in Figure 5), described in the previous section, to be applied in the routing process. For this reason the first proposal to apply the *PBR* to *WDM* networks considers one history register and a Prediction Table for every lightpath (route and wavelength). The history register keeps the history of the previous connection requests on that lightpath, and the Prediction Tables keep the information about connection blocking. After checking different options, the history register of the previous connection requests was defined by a vector holding a bit for each unit of time. This bit reflects if there was a connection established in that lightpath in that unit of time. The seed of this Thesis was this initial idea and it was proposed for *RWA* in *WDM* networks [21]. However, while the idea was being developed the performance evaluation results shown a more appropriate simple approach to bring the prediction concepts to routing in *WDM* networks. This simpler approach considers only a two-bit counter for route and wavelength, i.e., lightpath, without taking account the history registration. That is, this approach is more similar to the first branch prediction scheme reviewed in Section 3 that takes prediction decisions using the Program Counter, *PC*. Then in this simpler approach, the two-bit counter and thus the prediction are associated with the lightpath, not with the lightpath connection request history. Moreover, from the work done in *WDM* networks, the possibility to apply the mechanism to *IP/MPLS* networks was emerging.

One of the specific characteristics of the *PBR* mechanism is that it takes into account the previous blocked connections produced in the same lightpath. Usual *RWA* algorithms compute the lightpaths (route and wavelength) from the network state information obtained in the update messages. If the information is completely accurate the route decision will be the best. But when it exist certain degree of inaccuracy this network state information is not so useful. Moreover, these usual *RWA* algorithms do not take into account explicitly the past blocked connection on that lightpath. Just as an example, a connection is requested between a source and a destination node, the source node computes by means of a usual *RWA* algorithm the best path from the inexact network state information, and this connection request is blocked. In the case that immediately a connection is requested between the same source destination nodes and presuming there is not update of information between these two consecutive requests, the *RWA* algorithm would select the same lightpath. This usual *RWA* algorithm would not take into account the information stating that the previous connection request has been blocked when selecting the same lightpath.

The Thesis is organized in four parts, this part, Introduction; the second part is dedicated to Optical Networks (*WDM*); the third to *IP/MPLS* networks; and finally the fourth part concludes the Thesis. In the following paragraphs the different parts and their sections are briefly described.

Part II: *WDM* Networks

Sections 5 and 6

The Optical Network part reviews some of the recent work addressing the *RWA* problem taking and not taking into account the routing inaccuracy problem.

Section 7

The first approach to the *PBR* mechanism is presented in Section 7 (7.1 and 7.2) of this part considering the use of history registers and Prediction Tables with different entries. Then, due to the results obtained in the simulations, the initial idea was modified. An enhanced and simplified algorithm inferred from the *PBR* is presented in Section 7.3.c. where history registering is not needed and also the *PTs* have only one entry.

Section 8

Section 8 reviews the main concepts of hierarchical optical networks. Two new routing algorithms inferred from the *PBR* mechanism for hierarchical optical networks are proposed.

Section 9

Finally section 9 overviews some concepts of the Multilayer Traffic Engineering; and the *PBR* mechanism is proposed to be used in the optical layer of a Multilayer Traffic Engineering strategy.

Part III: IP/MPLS Networks

Sections 10 and 11

The third part of this Thesis is devoted to *IP/MPLS* networks. Section 10 of this part describes some of the previous works about *QoS* routing. Moreover in *IP/MPLS* networks there are proposed in the literature some routing mechanisms based on predictive concepts, reviewed in Section 11.

Section 12

The *PBR* mechanism applied to *IP/MPLS* networks has been developed from the initial ideas presented for optical networks, although some of the work has been done in parallel. Bringing the concepts of branch prediction to an *IP/MPLS* routing environment was done in the same way as in optical networks. There were some Prediction Tables and some registers, one for every route. In the first approximation to the problem, the bandwidth allocated on a route was considered the information to be registered. And the bandwidth requested by a connection added with the bandwidth yet allocated on the route was considered the information to build the index to access the Prediction Table. But this initial idea was changing from the results obtained in the simulations, and different routing algorithms were proposed. In Section 12 it is described the *PBR* mechanism for *IP/MPLS* networks and all the routing algorithms inferred from it.

All the different routing algorithms proposed in both parts are evaluated by means of simulations. The different simulators used were specially developed programming in C for this Thesis.

Part IV: Conclusions and Future Work

Section 13:

This section reviews and summarizes the proposed ideas of this Thesis. Moreover the main conclusions about the *PBR* mechanism are presented.

Section 14:

In this section some of the possible future work related to this Thesis is presented.

PART II

WDM NETWORKS

This part reviews some recent work addressing the Routing and Wavelength Assignment (*RWA*) problem in Wavelength Division Multiplexing (*WDM*) networks. It presents the Prediction-Based Routing Mechanism (*PBR*) as a new *RWA* mechanism for WDM networks. The new mechanism is explained by an illustrative example and evaluated by different simulations.

5. Routing and Wavelength Assignment in WDM Networks

Unlike traditional *IP* networks where the routing process only involves a physical path selection, in *OTN*s (Optical Transport Networks) the routing process not only involves a physical path selection process (i.e., find a route from the source to the destination node) but also a wavelength assignment process (i.e., assign a wavelength –or wavelengths- to the selected route), named the routing and wavelength assignment (*RWA*) problem. The *RWA* problem is often tackled by being divided into two different sub-problems, the routing sub-problem and the wavelength assignment sub-problem. Figure 6 shows a scheme of the *RWA* classification.

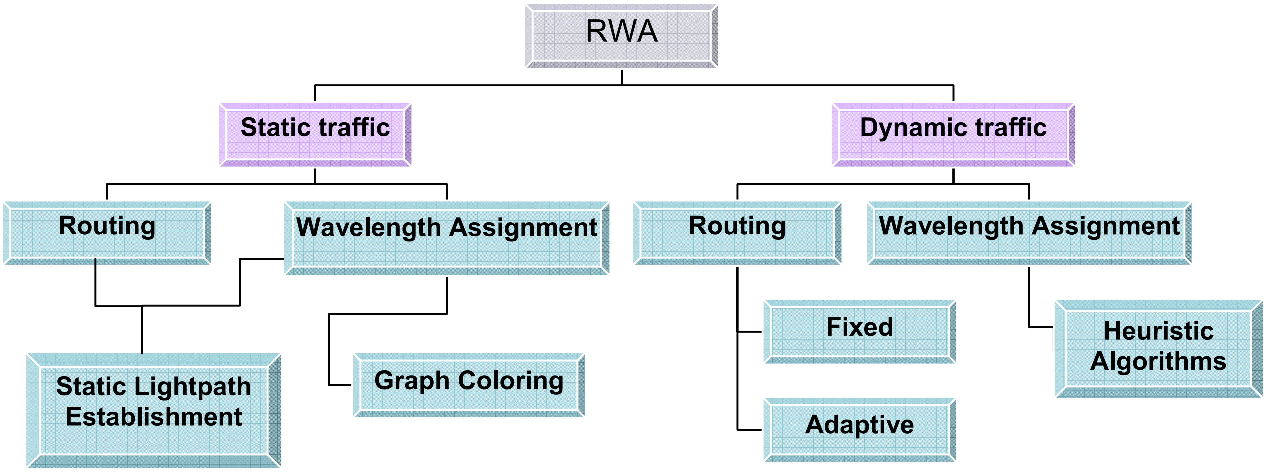


Figure 6. *RWA* Classification.

5.1. The *RWA* problem with static traffic

With static traffic, the entire set of connection requests is previously known, and the static *RWA* problem of setting up these connection requests is named the Static Lightpath Establishment (*SLE*) problem. The objective is then to minimize the network resources such as wavelengths or fibres required to establish these connection demands, or also in other words the objective can be to maximize the number of established connections among the entire set for a given number of resources, wavelengths and fibres. The *SLE* problem can be formulated as a mixed-integer linear program, such as Ramaswami et al presented in [22], which is NP complete. There are different proposals to solve the *SLE* problem, genetic algorithms or simulated annealing presented in [23] by Zhang et al, can be applied to obtain locally optimal solutions. In general, in order to make the *SLE* problem more tractable, it is divided into two subproblems, the routing subproblem and the wavelength assignment subproblem. For example in [24] Banejee et al propose to use LP (Linear Programming) relaxation techniques followed by rounded to solve the routing subproblem, and graph colouring to assign the wavelengths once the routes has been assigned.

Often, in this scenario, the *SLE* problem is also referred as the virtual topology problem [25][26].

5.2. The *RWA* problem with dynamic traffic

In a dynamic traffic scenario the connections are requested in some random fashion, and the lightpaths have to be set up as needed. Source-based routing is one of the recommendations stated in the *ASON* specifications [27]. According to the source-based routing, routes are dynamically computed in the source nodes based on the routing information contained in their network state databases. There are many contributions in the literature addressing the dynamic *RWA* problem and proposing some algorithms dealing with both the routing selection, and the wavelength assignment subproblems.

5.2.1. The Routing Subproblem

Concerning to the routing subproblem, the routing algorithms can be classified in two different classes: off-line (fixed) and on-line (adaptive). In off-line routing, the algorithm is executed off-line and the set of precomputed routes for every source-destination node pair are stored for latter use. An example is the shortest path (*SP*) algorithm. The main drawback of the *SP* algorithm is the lack of network load balance since the selected route between a fixed pair of nodes will always be the same regardless the traffic load. In [28] Harari et al propose the fixed-alternate routing algorithm which provides the network with more than one route for each pair of nodes. Unfortunately, off-line routing does not consider the current network state when computing routes, which significantly impacts on the global network performance. Instead, on-line (or adaptive) routing relies on the network state information when computing routes. These adaptive algorithms are executed at the time the connection requests arrives. In on-line routing, the route can be calculated reacting to a path request (i.e. on-line) or the routes can be precomputed (off-lines) being then the on-line algorithm which selects one of them according to the current network state information.

An example of these dynamic algorithms is the Least-Loaded Routing (*LLR*), presented in [29] by Chan et al, where the selected route is the less congested among a set of precomputed routes, that is the route with more available wavelengths. Congestion in a route is defined as the congestion of the most congested link on the route, that is, the link with less available wavelengths. Two variants of the *LLR* algorithm are proposed by Li et al in [30]. The first algorithm is called *FPLC* and is basically the same as the *LLR* but limiting the number of precomputed routes to the two shortest and link disjoint routes. The use of link disjoint routes is very usual in many *RWA* algorithms. The main reasons are that the algorithm will select among parallel routes, and also because, if one route fails the connection can be rerouted to another route. Authors in [30] argue that the use of more than two routes do not significantly improve the performance. The second proposed algorithm in [30] is the *FPLC-N(k)*. In this case, instead of searching for the availability of the wavelengths on all links of the precomputed routes, only the first k links on each route are searched. This solution tries to achieve a trade-off between low control overhead and low blocking probability.

On the other hand the algorithms proposed in [31] by Todimala et al compute the route dynamically instead of being selected among a fixed set of precomputed routes. These algorithms are the Least Congested Shortest Hop Routing (*LCSHR*) and the Shortest Hop Least Congested Routing (*SHLCR*). In the first one, *LCSHR*, the priority is to efficiently utilize the routes, and so it selects the least congested route among all the shortest hop routes currently available. In the second, *SHLCR*, the priority is to efficiently maintain the load in the network, and so it selects the shortest hop route among all the least congested routes.

5.2.2. The Wavelength Assignment Subproblem

The wavelength assignment process is valid for static traffic or for dynamic traffic. Usually the static wavelength assignment is solved by means of graph-colouring, for example by Mukherjee in [32]. On the other hand, there are several heuristic algorithms proposed in the literature dealing with the dynamic assignment problem, such as Random, First-Fit (*FF*), Least-Used (*LU*), Most-Used (*MU*) and Max-Sum (*MS*) (by Subramanian et al in [33]), Min-Product (*MP*) (by Jeong et al in [34]), Least-Loaded (*LL*) (by Karasan et al in [35]), Relative Capacity Loss (*RCL*) (by Zhang et al in [36]), Protecting Threshold and Wavelength Reservation (*Rsv*) (by Birman et al in [37]) and Distributed Relative Capacity Loss (*DRCL*)(by Zhang et al in [38]).

The Random (*R*) scheme randomly assigns a wavelength among all the available wavelengths on the route. The First-Fit (*FF*) scheme has numbered all the possible wavelengths. The wavelength selected is that with the low number among the available on the route. The Least-Used (*LU*) scheme selects the wavelength that is the least used in the network. The Most-Used (MU) the opposite of LU, it attempts to assign the most used wavelength in the network. This is done to pack the connections in fewer wavelengths. The Minimum Product (*MP*) is for multi-fibre networks, where the links between nodes consists in several fibres, and then there are several wavelengths of each colour. It tries to minimize the number of needed fibres in the network. First, for each wavelength the product of the assigned (or occupied) fibres on each link of the route is done. Then, the wavelength selected is that with the lower number among the wavelengths that minimizes that product. In a single-fibre network the number of possible assigned fibres in each link of the route only can be 0 (if it is free that wavelength) or 1 (if it is assigned). So, the product for the wavelengths that are available in the route will be 0, and the *MP* becomes the *FF*.

The Least-Loaded (*LL*) selects the wavelength with more capacity (more not assigned fibres) in the most loaded link of the route. Like the *MP* scheme is designed for multi-fibre networks and also becomes the *FF* in single-networks.

The Max-Sum (*MS*) scheme is designed for both, single and multi-fibre networks. It considers all possible lightpaths (route and wavelength) between a source and destination node. It selects the wavelength that will maximize the sum of remaining capacities (free fibres, or not assigned) of all the other lightpaths. That is, the Max-Sum scheme selects the wavelength that minimizes the capacity loss due to set up a lightpath.

Similar to the Max-Sum the Relative Capacity Loss (*RCL*) scheme bases its decision on selecting that wavelength minimizing the relative capacity loss due to set up a lightpath with this wavelength.

The schemes Wavelength Reservation (*Rsv*) and Protecting Threshold (*Thr*) try to protect long routes instead of minimizing the blocking probability. Applying them, the long routes will not suffer high blocking probabilities, achieving a greater degree of fairness. The complete fairness is achieved when the blocking probability is independent of the source, destination nodes and number of hops of the route. That is, all the routes suffer the same blocking probability, independently of the length. The Wavelength Reservation scheme reserves wavelength in those links to be used only by long routes that traverses that link. In the case of Protecting Threshold, a wavelength is assigned to connections of single-hop only if there is a minimum value (threshold) of free wavelengths.

A variant of the Relative Capacity Loss (*RCL*) is the Distributed Relative Capacity Loss (*DRCL*) which is applied for online calculation of routes while RCL is applied for fixed routes.

It is necessary to mention that most of the routing algorithms reviewed in subsection 5.2.1. are combined with some of the wavelength assignment algorithms described above. Usually, first the routing algorithm selects a route and then the wavelength algorithm selects a wavelength among those available for such a route. Just as an example, the routing algorithms *LCSHR* and *SHLCR* [31] are combined with the First-Fit (*FF*) and Most-Used (*MU*) schemes of wavelength assignment to evaluate the blocking probability produced by such combinations.

There are other techniques such as the unconstrained routing presented in [39] by Mokhthar et al, where the route is selected once the wavelength has already been assigned. Hence, firstly the wavelengths are ordered according to their use and the most used (*MU*) wavelength is selected, and then the shortest route on this wavelength is dynamically computed.

5.3. The *RWA* problem in Wavelength Interchangeable Networks.

In general, to establish a lightpath, that is, to select a route and to assign a wavelength on the selected route, it is required that the same wavelength will be used on all the links in the end-to-end route. This constraint is known as the wavelength continuity constraint. Wavelength routed networks without wavelength conversion are known as Wavelength-Selective (*WS*) networks. Networks under this constraint exhibit poor results in global network blocking. In order to improve the network performance the wavelength continuity constraint can be eliminated by introducing wavelength converters. Wavelength routed networks with wavelength conversion are known as wavelength-interchangeable (*WI*) networks. In such networks, the Optical Cross-Connects (*OXCs*) are equipped with wavelength converters so that a lightpath can be set up using different wavelengths on different links along the route. It is widely shown in the literature the positive effects in the network performance because of adding wavelength conversion capabilities (see for example Kovacevic et al [40]. and Ramamurthy et al [41]).

If all the *OXCs* of the network are equipped with wavelength converters it is referred such as full wavelength conversion. When full wavelength conversion is available at all nodes the *WDM* network is equivalent to a circuit-switched network. Unfortunately, wavelength converters are still very expensive. If only a percentage of the *OXC* has wavelength converters it is referred such as sparse wavelength conversion. There are many proposals to allow the network to include wavelength conversion capabilities also minimizing the economical cost by means of sparse wavelength conversion.

Many of the reviewed *RWA* algorithms for WS networks do not consider explicitly the length of the routes in the route selection. In this *WS* networks usually shortest routes are those having more available wavelengths, since the probability of longer routes with a large number of available wavelengths is very low. However this property is carried out only weakly in *WI* networks. For this reason, usual *RWA* algorithms for *WI* networks take into account explicitly the length of the route in its decision.

In [42][43] Chu et al present a *RWA* algorithm for networks with sparse wavelength conversion, the Weighted Least-Congestion Routing-First-Fit (*WLCR-FF*), in conjunction with a simple greedy wavelength converter placement algorithm. The *WLCR-FF* algorithm selects the route maximizing the weight $\frac{F}{\sqrt{h}}$ among a set of precomputed shortest and link disjoint routes. The parameter F accounts for the availability of the route, being the number of common wavelengths on all the links of the route for *WS* networks. Instead, for WI networks with full wavelength conversion, F is the smallest value of available wavelengths among the links of the route. And finally, for sparse wavelength conversion, F is the smallest value of available wavelengths among all the segments of the route between wavelength converters. The parameter h is defined as the length of the route in number of hops. Once the route is selected, the First-Fit algorithm is applied in every one of the segments of the route to select the wavelengths.

Masip et al in [44] present an algorithm, ALG3, based on the *BBOR* mechanism that also selects a route among a set of precomputed shortest and link disjoint routes. However this algorithm selects both the route and wavelength simultaneously, that is the lightpath. First, the algorithm can select among a set of previously computed routes. Then, the algorithm calculates a weight for every lightpath, that is, for every combination of precomputed route and possible wavelength. This weight is n·(L/F), being *L* the number of links of the lightpath where that wavelength has been defined as Obstruct-Sensitive-Wavelength (*OSW*). A wavelength is defined as *OSW* in a link when the number of available wavelengths of that colour is lower or equal to a percentage of the number of changes (threshold) needed in the network state to send an update message. This threshold value is established by the triggering policy. *F* is the minimum value of available wavelengths of that colour along the links of the lightpath. The length of the path in number of hops, *n*, is included to avoid selecting long paths. *L* represents the degree of obstruction of the lightpath and *F* represents the degree of congestion. Note that the definition of *F* differs from the definition exposed in the previous *WLCR-FF* algorithm. In the *WLCR-FF*, there is an *F* value for every route. In that case, *F* is the minimum number of common wavelengths of different colours in the different links or segments along the route. However in the *BBOR* mechanism there is an *F* value for every lightpath (route and wavelength). *F* is the minimum number of available wavelengths of one colour in the links along the route. The *BBOR* mechanism aims to compute the lightpaths taking into account the inherent inaccuracy of the network state information. The main concepts of the *BBOR* mechanism are reviewed in Section 6.

5.4. Other *RWA* techniques

In [45] Zhou et al proposed that the state of a multifibre link is given by the set of free wavelengths in this fibre and is represented as a compact bitmap. For every source-destination pair of nodes and every fibre, there is an *n*-bit integer variable used to keep track of the free wavelengths in this fibre; being *n* the number of wavelengths. Every position in this *n*-bit integer variable only can hold a 1-value if that wavelength is freed (available) or a 0-value if it is occupied. Then, the state of a lightpath is represented by a similar bitmap computed as the logical intersection of individual bitmaps of the links of the path. The count of number of bits with 1-value in the bit map of the path is used as the primary gain function in the path selection. Authors developed a modified Dijkstra's algorithm that takes into account this gain function to compute the shortest cost path.

It is worth mentioning that there are other different approaches to solve the *RWA* problem. There are some proposals addressing the problem by means of genetic algorithms, for example [46] by Bisbal et al and [47] by Le et al Other works utilizes the notion of ant agents or ant colony, in [48] by Garlic et al and in [49] by Le et al; or even the combination of both, ant agents and genetic algorithms, in [50] by Le et al.

6. The Routing Inaccuracy Problem.-State of the Art.

Most of the reviewed dynamic *RWA* algorithms assume that the network state databases contain accurate network state information. Unfortunately, when this information is not accurate enough, the routing decisions taken at the source nodes could be incorrectly performed hence producing a significant connection blocking increment (the routing inaccuracy problem comes up). The routing inaccuracy problem concerns to the impact on global network performance when taking *RWA* decisions according to inaccurate (or outdated) routing information. In highly dynamic networks, inaccuracy arises mainly due to the restriction to aggregate routing information in the update messages, the frequency of updating the network state databases and the latency associated with the flooding process. It is worth noting that the first two factors attempt to reduce the signalling overhead.

The most recent studies dealing with the routing inaccuracy problem in optical networks can be found in [51]-[60], and [44]. The contributions in [51]-[56] evaluate the impact on the blocking probability because of selecting lightpaths under inaccurate routing information. The proposed analytical models and the presented simulation results show that the blocking ratio increases in a fixed topology when routing is done under inaccurate information. To counteract this blocking effect, new Routing and Wavelength Assignment (*RWA*) algorithms, able to tolerate inaccurate network state information have been proposed in [57]-[60],[44].

Most of them deal with wavelength switched networks without wavelength conversion capabilities (that is wavelength selective networks, *WS*) and not much deal with networks with wavelength conversion capabilities (that is wavelength interchangeable networks *WI*).

Jue et al in [51] present for the first time an analytical model to evaluate the blocking caused by the routing inaccuracy problem. This work is significantly enhanced in [52] by Lu et al The proposed model includes two kinds of traffic blocking: that caused by insufficient network resources and that caused by outdated information. Assuming fixed routing (shortest path), random wavelength selection and no wavelength conversion, the authors carried out some simulations on the PacNet network to verify the accuracy of the proposed model. After comparing the analytical results to the obtained simulation results they conclude that the analytical results are highly accurate under both light and heavy traffic load.

In [53] Zhou et al present some simulation scenarios to show the negative effects produced in the connection blocking probability because of selecting paths under inaccurate routing information. The authors indeed verify over a fixed topology that the blocking ratio increases when routing is done under inaccurate routing information. The routing inaccuracy is introduced by applying an update policy based on time, so that the network state databases are updated according to an update interval of 10 seconds. Therefore, it is possible that the wavelength selected by the source node for a source-destination node pair at the path selection time will not be available at the path setup time resulting in the blocking of the connection. Some other simulations are also performed to show the effects on the connection blocking probability because of changing the number of fibres on all the links. Finally, as a conclusion, the authors argue that new routing algorithms tolerating inaccurate global network state information must be developed for dynamic connection control/management in *WDM* networks.

In [57] Zheng et al assume that distributed routing based on global network state information requires strict guarantees in the routing information accuracy. To reduce the inaccuracy, authors assume that the routing information is updated whenever there is a change. However, as stated before, the non-negligible propagation delay also yields to outdated information. Therefore, authors propose a distributed lightpath control scheme based on destination routing in order to select paths based on the most recent network state information. The mechanism is based on both selecting the physical route and wavelength on the destination node, and adding rerouting capabilities at the intermediate nodes in order to avoid blocking a connection when the selected wavelength is no longer available at the setup time at any intermediate node. In this work the information used by the destination node to select the lightpath is not collected by the setup message sent by the source node along the path but the information contained in the network state database of such a destination node. There are two main weaknesses of this mechanism. Firstly, since the rerouting is performed in real time in the setup process, wavelength usage deterioration is directly proportional to the number of intermediate nodes that must reroute the traffic. Secondly, the signalling overhead is not reduced, since the routing and wavelength assignment decision is based on the global network state information maintained on the destination node, which must be perfectly updated.

Another contribution on this topic can be found in [58] where Darisala et al propose a mechanism whose goal is to control the amount of signalling messages flooded throughout the network. Assuming that update messages are sent according to a hold-down timer regardless of frequency of network state changes, authors propose a dynamic distributed bucket-based Shared Path Protection scheme. This means that the amount of signalling overhead is limited by both fixing a constant hold-down timer which effectively limits the number of update messages flooded throughout the network and using buckets which effectively limits the amount of information stored on the source node, i.e. the amount of information to be flooded by nodes. The effects of the introduced inaccuracy are handled by computing alternative disjoint lightpaths which will act as a protection lightpaths when resources in the working path are not enough to cope with those required by the incoming connection. Authors show by simulation that inaccurate database information strongly impacts on the connection blocking. This connection blocking increase may be limited by properly introducing the suitable frequency of update messages. According to the authors, simulation results obtained when applying the proposed scheme along with a modified version of the *OSPF* protocol, may help network operators to determine that frequency maintaining a better trade-off between the connection blocking and the signalling overhead.

Solutions presented so far only tackle the routing inaccuracy problem in *WS* networks, i.e., in networks without wavelength conversion capabilities. Lu et al in [54] present an extension of the analytical model proposed in [53] to evaluate the blocking probability in wavelength switched networks with sparse wavelength conversion. In order to validate the proposed blocking model authors compare the analytical results to those obtained by simulations carried out on the PacNet considering fixed-shortest path routing and random wavelength converters placement. Summarizing what is the last contribution of these authors, they analyze the blocking probability taking into account three types of blocking, due to insufficient network resources, due to outdated information and due to over-reservation. The proposed models are evaluated and validated in comparison with the results obtained by simulating the fixed-shortest path routing with random wavelength selection on both the PacNet and a 12-node optical ring.

The routing inaccuracy problem is named wavelength contention in [59]. In this work, Lu et al initially review the problems involved because of the routing inaccuracy problem, that is, the over-reservation problem and outdated information problem respectively. Then they propose a new distributed signalling scheme named the Intermediate-node Initiated Reservation (*IIR*) to deal with both problems. Authors extend the analytical models already developed in [53] to evaluate the blocking probability in two main points: first they present a model in which reservations could be initiated by some intermediate nodes; second, they extend the model to be applied to networks with and without conversion capabilities. The main concept underlying this lightpath control scheme boils down to allow the reservation to be initiated by a set of intermediate nodes before the connection request reaches the destination node.

Contributions presented so far focus on mono-fibre wavelength routed networks. In the work presented in [55] Shen et al only show that the routing inaccuracy problem also exists in multifibre wavelength switched networks. Assuming source routing they analyze the routing blocking (due to insufficient resources) and the setup blocking (due to the routing inaccuracy problem). By running several simulations they measure the impact of the update interval on the blocking probability assuming adaptive shortest path routing on a 2-fibre wavelength routed network without conversion capabilities. They conclude that the impact of the routing inaccuracy problem on the global blocking probability depends on the traffic load. Also concerning multifibre wavelength routed networks and even though the proposal does not take into account the routing inaccuracy problem as a source of blocking. Lu et al-in [56] present an analytical model of the blocking probability for dynamic lightpath establishment also including an analysis of the model complexity.

The BYPASS Based Optical Routing *(BBOR)* proposed by Masip et al aims at reducing the connection blocking probability caused by taking routing decisions under inaccurate network state information in multifibre wavelength switched networks with [44] and without [60] wavelength conversion capabilities. The *BBOR* mechanism allows several nodes along the selected path to dynamically reroute the setup message in those links where there is no wavelength availability. The unavailability of the selected wavelength is produced by selecting the lightpath with inaccurate information. The *BBOR* mechanism consists on three steps: (1) Decide which wavelengths of which links (bundle of *B* fibres) need to have computed a bypass-path, (2) Select the lightpath using the information about the wavelengths that have to be bypassed. (3) Select the bypass-paths.

The wavelengths that need bypass-paths are defined as Obstruct-Sensitive-Wavelengths (*OSW*). A wavelength in a link is considered *OSW* depending on the triggering policy. The triggering policy proposed in the *BBOR* mechanism is as follows. A node sends an update message whenever there are *N* changes in the network state, that is, *N* lightpaths are set up or torn down. Being *B* the number of fibres on the link, *B* is also the total number of wavelengths of each colour, and *R* the number of currently available of those wavelengths, a wavelength is defined as *OSW* when *R* is lower or equal than a percentage of the threshold value of updating, *N*.

In [60] authors propose two algorithms, ALG1 and ALG2, which take into account the number of links where a wavelength has been defined as *OSW* in order to compute the lightpath, for networks without conversion capabilities. ALG1 and ALG2 assume that *L* is the number of links of the lightpath where that wavelength has been defined as *OSW*, and *F* is minimum value of available wavelengths of that colour (number of fibres where that wavelength is available) along the links of the lightpath. *L* accounts for the obstruction and *F* for the congestion. Firstly, both algorithms compute the shortest available path. ALG1 selects that lightpath that minimizes *L*, that is, the number of links where the wavelength is *OSW*. If more than one lightpath exists the less congested is selected, that maximizing *F*.

However ALG2 selects that lightpath among the shortest available that maximizes *F*, that is, the less congested. If more than one exist selects that minimizing *L*.

Once the lightpath is selected the bypass-paths has to be computed for each link on the lightpath where the wavelength has been defined as *OSW*. The shortest bypass paths are computed. When an intermediate node in the lightpath selected detects a link without available wavelength would reroute the setup message along the computed bypass-path.

ALG3 is proposed in [44] for *WI* networks, but it can be implemented for networks without conversion capabilities. This algorithm has been reviewed in the previous section, Routing and Wavelength Assignment in *WDM* Networks. It selects the lightpath that minimizes the weight n·(L/F) among the k-shortest and link disjoint paths. Once the lightpath is selected the corresponding bypass-paths are also computed.

7. The Prediction-Based Routing Mechanism in Flat WDM Networks.

7.1. Motivation

One of the *ASON* recommendations focuses on *RWA* solutions based on distributed source-routing. In this scenario the routing inaccuracy problem comes up. As it is explained in a previous section the routing inaccuracy problem describes the impact on global network performance because of taking *RWA* decisions according to inaccurate or outdated routing information. It has been clearly shown [53] that the routing inaccuracy problem, may have a significant impact on global network performance in terms of connection blocking.

The Prediction-Based Routing (*PBR*) is aimed to reduce both the signalling overhead and the negative effects of the routing inaccuracy problem. The main concept of the *PBR* mechanism boils down to select routes not based on the 'old' or inaccurate network state information but based on the history of previous connection requests.

The Prediction Based Routing *(PBR)* mechanism is based on extending the concepts of branch prediction presented by Smith in [61] and used in the computer architecture area. In this field, there are several methods to predict the direction of the branch instructions. The prediction of branch instructions is not done knowing the exact state of the processor but knowing the previous branch instructions behaviour. There is a detailed explanation of the basic concepts used in branch prediction in Section 3 of this Thesis. Bringing the branch prediction concepts to a network scenario, the *PBR* mechanism is based on predicting the lightpath, that is, the selected route and the assigned wavelength between a source-destination node pair according to the routing information obtained in previous connections requests. Thus, the *PBR* mechanism does not need the network state information obtained from the network state databases to compute the lightpath. As a consequence, the frequent flooding of update messages is substantially reduced (only minimal updating is required to ensure connectivity).

7.2. Description and Data Structures.

The main objective of the *PBR* mechanism is to optimize the routing decision not using the network state information but taking into account the history of each lightpath. Next subsections clearly describe the *PBR* mechanism.

A. History Registration

Assuming source routing, the method used to register the history of the network state is based on keeping in every source node a history for every wavelength and path (for every lightpath) and destination. This lightpath history includes the information about when a connection was established previously in that lightpath. Every lightpath history is stored in a history register named Wavelength Register (*WR*), holding a vector of 0s and 1s reflecting this history. In the source nodes there will be one of such registers for every wavelength on every path (for every lightpath) to every destination node.

As it is mentioned above the *WRs* are vectors of 0s and 1s. Every unit of time the *WR*s are modified by means of shifting the vector one position to the left and setting a new value on the right. A unit of time is the time value used to measure the simulations timing, including holding time, arrival time, and time between updating. Each *WR* is updated setting a 0 value whenever this lightpath is used on that unit of time. Otherwise, the register of an unused lightpath is updated setting a 1. It must be noticed that the expression "a path is used" means that a connection is established in that path. On the other hand, "a path is unused" when no incoming connection is assigned to this path.

In Figure 7 there is an example of *WR* for a particular lightpath, containing information about the last 12 units of time. It is assumed the value on the right as the newest and the value on the left as the oldest. Thus, for instance looking at Figure 7, whereas there was a connection established in that lightpath on the last two units of time, there was not a connection established three units of time ago.

B. Prediction Tables

The *WR*s are used to both train and index new defined tables, named Prediction Tables (*PT*). These *PT*s have different entries, each keeping information about a different pattern

0	1	0	0	0	1	1	1	1	1	0	0

Figure 7. Example of Wavelength Register, *WR*.

by means of a counter. One *PT* is needed in the source nodes for every feasible lightpath between any source-destination node pair. For example, assuming that a source node sends traffic towards two different destination nodes through two different routes (assuming for instance the two shortest-paths), and with 6 wavelengths per route, then 24 *PTs* are needed on the source node, that is, one *PT* for every path and wavelength. In every source node, there is the same number of *WR*s than *PT*s. The *PT* for a wavelength on a route is accessed by an index which is obtained from the corresponding *WR*. The indexes built from the *WR*s have information about the last and previous units of time so that the information about the current unit of time is not included. This statement is justified because while the occupation of wavelengths can change along the current unit of time, i.e., new connections are setup or existing connections are torn down, the *WR*s are only updated once per unit of time.

Every entry in the *PT*s has a counter, which is read when accessing the table. The obtained value is compared to a certain threshold value. If the value obtained after reading the *PT* is lower than the threshold, the prediction is to accept the request through this wavelength on this route. Otherwise, the path is predicted to be unavailable. The threshold value depends on the number of bits used for the counter. The counters are two-bit saturating counter, where 0 and 1 stand for the lightpath availability and 2 and 3 stand for the lightpath unavailability. Saturating counter means that the counter value does not change when decreasing from a value of 0, nor when increasing from a value of 3. The use of two values to account for the availability or unavailability has been widely studied in the area of branch prediction on computer architecture [61].

As presented in [61] a two-bit counter gives better accuracy than a one-bit counter. The use of a one-bit counter means that it predicts what happened last time. In this case, if in the last time the traffic request was blocked then the next time that the history is repeated the prediction will turn out unavailability. Besides, if in the last time the traffic request was accepted the prediction will turn out availability. Instead, if the counter has two bits it is necessary that the traffic request had been blocked (or accepted) two times for the same history to change the direction of the prediction. It is also exposed in [61] that going to counters larger than two bits does not necessarily give better results. This is due to the "inertia" that can be built up with a large counter. In that case more than two changes in the same direction are necessary to change the prediction.

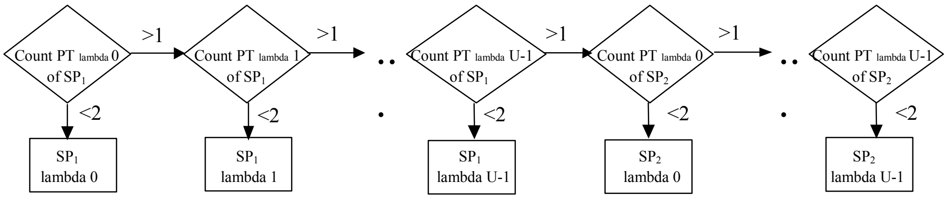


Figure 8. *RWP* flow chart.

The process of updating the *PTs* (i.e. training) is the following. When a new connection request is set up, the *PT* of the selected wavelength and path is updated, decreasing the counter. On the other hand, when the path and wavelength is selected but the connection request is blocked the counter is increased. Other *PT*s of the unselected paths are not updated.

It is worth noting that the updating of *PT*s in the source nodes is done immediately after the connection request is either set up or blocked. For this reason it is not necessary to flood update message throughout the network to update the network state databases.

7.3. Routing Algorithm Inferred from the *PBR* Mechanism for Wavelength Selective (WS) Networks.

A. Routing Algorithm for monofibre networks

Based on the *PBR* mechanism a new *RWA* prediction algorithm is defined, named Route and Wavelength Prediction *(RWP)* algorithm [21], which utilizes the information contained in the *PTs* to decide about which path and which wavelength will be selected. The *RWP* performs as follows. When a new request arrives at the source node demanding a connection to a destination node, all the *PT*s of the corresponding destination are accessed. It must be noticed that one *PT* and one *WR* exist for every wavelength on every path to every destination node. It is assumed that two shortest paths are computed for every source-destination node pair, SP_1 and SP_2. These shortest paths are link disjoint if possible, otherwise the shortest paths should share the minimum number of links. The casue motivating this is based on the fact that when a route (*SP1* or *SP2*) is predicted to be blocked, the source node does not know the link blocking the route.).

The *PT*s are accessed by one index per table which is built from the corresponding *WR*. In Figure 8 it is presented a flow chart depicting the *RWP* performance assuming *U* wavelengths in every link. The *RWP* algorithm always starts by considering the value of the counter of the *PT* of the first wavelength on the first shortest path, for instance *SP_1*. If the counter is lower than 2 (0,1) and this wavelength is available in the node's output link towards *SP_1*, the prediction algorithm decides to use this wavelength on this path. Otherwise (counter=2, 3 or output link not available) this wavelength is not used. In this last case, the value of the counter of the next *PT* is examined. The next *PT* corresponds to the second wavelength on *SP_1*. The information about the current unit of time in the prediction decision is introduced by the output link availability. This information along with the *PT*s counter is the information checked by the *RWP* algorithm. Once the counters of the *PT*s of all the wavelengths of *SP_1* have been examined, (that is, either the counters always are greater than 1 or all wavelengths on the output link towards *SP_1* are not available), the prediction algorithm checks the *PT*s of the next path, *SP_2*.

Being aware that every source node knows its output link availability, as a last option before blocking the incoming connection (when the prediction algorithm, after checking all *PT*s, decides that all the feasible wavelengths on both paths are predicted to be blocked) the source node tries to forward the connection request through the first available wavelength on the output link towards one of the two shortest paths. The attempt of selecting the routes by just checking the output availability when no lightpath can be assigned is done to unblock the *PT* counters. Indeed, when neither path nor wavelength is selected (because all *PT* counters are larger than 1), the *PBR* mechanism assigns the request to the first available wavelength on the output link towards *SP_1*. If the path can neither be assigned, then the algorithm assigns the request to the first available wavelength on the output link towards *SP_2*. If the path and wavelength can be selected by means of this method, and the connection can be established, then the corresponding *PT* counter of the corresponding wavelength of *SP_1* or of the *SP_2* is decreased, hence unblocking it. If there is not any available wavelength in any output link for both shortest paths the incoming connection is finally blocked.

As stated in a previous subsection, the *WR*s are updated every unit of time according to the wavelengths and paths which are used. The *PT* of the selected wavelength and path is also updated by either increasing (means connection blocked) or decreasing (means connection not blocked) the counter of the corresponding entry in the *PT*.

```
1.  Order(Route SP1)
      ($o_0$, $o_1$……$o_{U-1}$ is the index wavelength order for checking Route SP1)
2.  Check(Route SP1):
      i=0;
      while (route is not assigned and i<U){
            if (PTcounter($o_i$)<2 and wavelength $o_i$ is available in output link to route SP1)
                  { assign route SP1 and wavelength $o_i$;
                        if connection is established decrease PTcounter($o_i$)
                        else increase PTcounter($o_i$)
            }endif
            i++;
      }endwhile
3.  If (route is not assigned) {
4.  Order(Route SP2)
      ($o_0$, $o_1$……$o_{U-1}$ is the index wavelength order for checking Route SP2)
5.  Check(Route SP2):
      i=0;
      while (route is not assigned and i<U){
            if (PTcounter($o_i$)<2 and wavelength $o_i$ is available in output link to route SP2 )
                  { assign route SP2 and wavelength $o_i$;
                        if connection can be established decrease PTcounter($o_i$)
                        else increase PTcounter($o_i$)
            }endif
            i++;
      }endwhile
    }endif
6.  If (route is not assigned){
7.  CheckF(Route SP1):
      i=0;
      while (route is not assigned and i<U){
            if (wavelength i is available in output link to route SP1)
                  { assign route SP1 and wavelength i;
                        if connection is established decrease PTcounter(i)
                        else increase PTcounter(i)
            }endif
            i++;
      }endwhile
    }endif
8.  If (route is not assigned) {
    CheckF(Route SP2):
      i=0;
      while (route is not assigned and i<U){
            if (wavelength i is available in output link to routeSP2)
                  { assign route SP2 and wavelength i;
                        if connection is established decrease PTcounter(i)
                        else increase PTcounter(i)
            }endif
            i++;
      }endwhile
    }endif
```

Figure 9. Pseudo-code of the *RWP*-o algorithm.

B. Routing Algorithm for multifibre networks

Up to now, the *RWP* description only considers one fibre per link. However, the algorithm can be enhanced when assuming *n* possible fibres. Although the algorithm always checks *SP*$_1$ and *SP*$_2$ in this order, the algorithm can check the *PT*s (of each wavelength per path) according to two different policies. The first policy considers that the *PT*s are checked in a fixed order according to the number assigned to each wavelength. In this case the proposed algorithm is named *RWP-f*. The *RWP-f* algorithm selects the first lightpath accomplishing that its two-bit counter is lower than 2 and having output link availability. Under the second policy the wavelengths for each route are ordered according to the number of available fibres on each wavelength. In this case the algorithm is named *RWP-o*. That is, the *RWP-o* algorithm selects the lightpath with more available fibres (less loaded) among the lightpaths with their two-bit counter lower than 2 and output link availability It is important to note that the information about the number of available fibres for every wavelength used to order the *PT*s is that known by the source node (local information), which certainly might not be accurate since update message have been removed. The *PT*s are hence checked according to one of the two policies explained above. The decision of which wavelength and route are selected is done depending on the value of the counters of the *PT*s and the availability of the node's output links. Just as an example, in Figure 9 it is showed the core of the pseudo-code of the *RWP*-o algorithm. In short, the wavelengths of route SP_1 are checked (Routine Check(Route *SP1*)). If the algorithm does not select any wavelength in route SP_1, then route SP_2 is checked (Routine Check(Route *SP2*)). Afterwards, if there is not yet assigned wavelength and route in SP_1 nor SP_2, the algorithm tries to assign the wavelength in route SP_1 only checking the availability of the node's output link (Routine CheckF(Route *SP1*)). If the algorithm still has not assigned any route, it tries to assign (Routine CheckF(Route *SP2*)) the wavelength in route SP_2 only checking the availability of the node's output link. Otherwise the connection will be blocked.

C. *Routing algorithm simplification*

The algorithm enhancement [62] described in this subsection focuses on showing that the information about the last and previous units of time required so far is not needed. This means that the *WRs* are no needed so that *PT*s of only one entry (i.e., only one two-bit counter per route and wavelength) are enough to implement the *PBR* mechanism. The fact of removing the information about the last and previous units of time makes the *PBR* mechanism regardless of the unit of time selection. This enhancement will be justified by means of several simulations. Now, the two-bit counter can be interpreted as follows: the value of the counter for a route and wavelength is approximately the number of blocked connections produced the last two times that this route and wavelength was selected. A particular wavelength and route will not be selected (i.e., predicted to be blocked) whenever two blocking occur the last two times it was selected (counter>1). Instead, this route and wavelength will be selected whenever there is one blocking at top in the last two times it was selected (counter<2).

There is a two-bit counter per route and wavelength in the source nodes for every destination node. Just as an example, if a source node can forward connection requests to 2 different destination nodes through 2 possible routes for every destination, SP_1 and SP_2, and 4 possible wavelengths, then there are 16 two-bit counters in the source node. These two-bit counters are named as Wavelength Route Counters, *WR*C. The enhanced algorithm runs as shown in Figure 9 (notice that the *PT*s are only two-bit counters). Summarizing, for every new connection request, only the *WR*C values and the output link availability are checked according to the number of available fibres per wavelength (for example in *RWP*-o). The *PBR* mechanism becomes more scalable with this enhancement since only a two-bit counter is needed in the source nodes for every possible destination, route and wavelength.

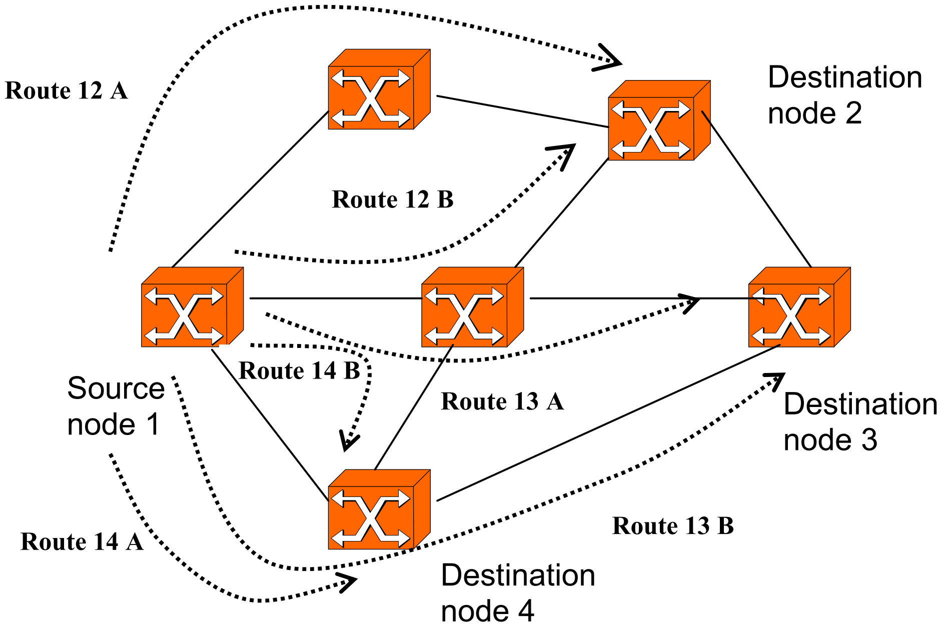


Figure 10. Topology used in the illustrative example.

7.4. Illustrative Example

Before evaluating the proposal, an example of the proposed algorithm is presented to illustrate its performance. Figure 10 shows the example topology being n1 a source node and n2, n3 and n4 destination nodes. Moreover, it is assumed that a link consists of one fibre with two wavelengths. In the figure we can see that there are two possible paths from the source node to each destination node, named 12A (i.e. source: n1, destination: n2, path: A), 12B, 13A, 13B, 14A, 14B. In node n1, there are 12 *WRCs*: WRC12AL1 (i.e., source: n1, destination: n2, path: A and L1: wavelength 1) WRC12AL2, WRC12BL1, WRC12BL2, WRC13ALl, WRC13AL2, WRC13BLl, WRC13BL2, WRC14AL1, WRC14AL2, WRC14BL1, WRC14BL2. Below, the evolution of the connection requests during 6 units of time is described.

Unit of time 1: Assuming that no more connections are established between n1 and any destination, a new connection request between n1 and n4 reaches n1 with a holding time of 4 units of time. Figure 11.a) shows both how the counters are read and how the prediction process works. Suppose that the algorithm orders the wavelengths according to the link availability turning out L2 and L1 for Route A and L1 and L2 for Route B. Remember that the algorithm orders the wavelength according to limited information only including the information known by the source node. The algorithm runs as follows. First, it checks the

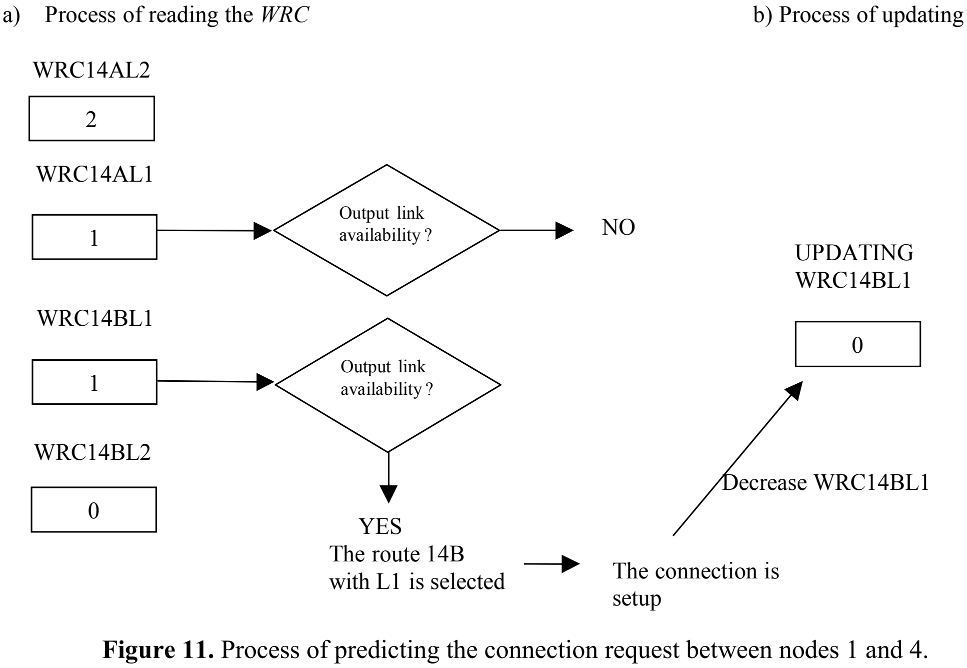


Figure 11. Process of predicting the connection request between nodes 1 and 4.

counter and the output link availability of route A and L2. The counter WRC14AL2 is 2 so that the prediction is that the connection will be blocked being this route and wavelength not selected. Afterwards, the algorithm checks the WRC14AL1 and the output link availability of route 14A with L1. This wavelength on this route is not selected since the output link is not available. Then, the algorithm checks route B. Since the counter WRC14BL1 is lower than 2 and the output link is available, then the prediction is that route B and L1 will not be blocked and hence are selected. In Figure 11.b) it is showed the updating of the *WRCs* for path 14B with lambda 1, WRC14BL1. The connection is set up without blocking and the WRC14BL1 is immediately updated, decreasing the counter.

Unit of time 2: No new connections are requested.

Unit of time 3: A new connection between node 1 and 2 is requested with a holding time of 3 units of time. The algorithm orders the wavelengths of path A, as L1, L2, and the wavelengths of path B as L2, L1. The path 12A with wavelength 1 is predicted to be available but the connection request is blocked. Figure 12.a) shows the prediction process. The counter WRC12AL1 is immediately updated hence being increasing (see Figure 12.b)).

Unit of time 4. No new connections are requested

Unit of time 5. In this unit of time there are not new connection requests. However it is worth mentioning that the request between nodes 1 and 4 produced in unit of time 1 releases its links because the holding time has finished.

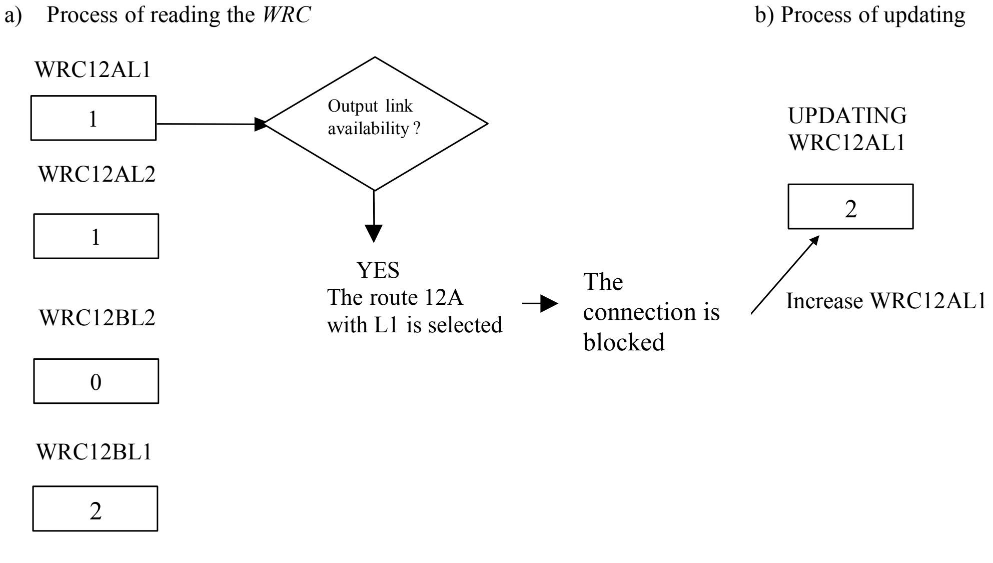

a) Process of reading the *WRC*
b) Process of updating
WRC12AL1
1
Output link availability ?
WRC12AL2
1
WRC12BL2
0
WRC12BL1
2
YES
The route 12A with L1 is selected
The connection is blocked
UPDATING WRC12AL1
2
Increase WRC12AL1

Figure 12. Process of predicting the connection request between nodes 1 and 2.

<u>*Unit of time 6*</u>. In this unit of time there are not new connection requests. The request between nodes 1 and 2 produced in unit of time 3 does not need to release its links because the connection was not established.

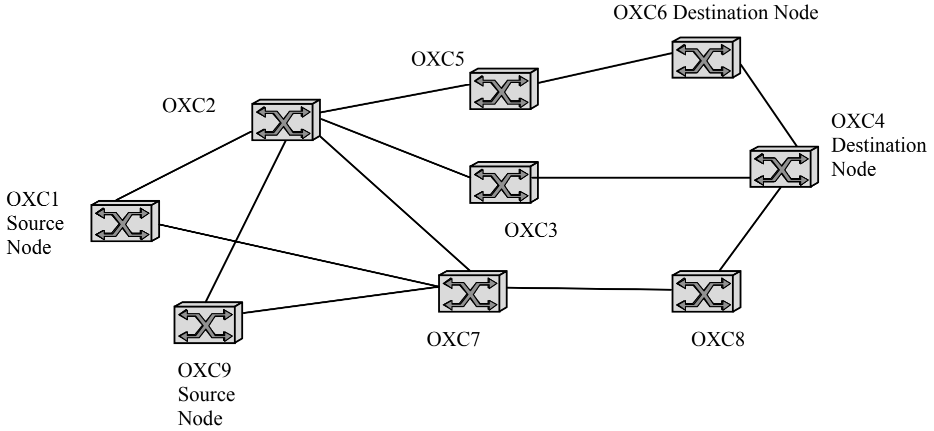
7.5. Performance Evaluation

7.5.1. Preliminary Evaluation

Once the proposed algorithm has been analyzed by the illustrative example presented in subsection 7.4, the performance of the *PBR* mechanism is evaluated on different network scenarios. First a preliminary evaluation of the *PBR* behaviour is carried out, analyzing the effect of different parameters, such as the number of *WR*s bits or the number of fibres and wavelengths. The *RWP* algorithm is compared with a well known routing and wavelength assignment algorithm, Shortest-Path combined with First-Fit for monofibre and combined with Least-Loaded for multifibre networks. That is, the route selected is the shortest available, and the wavelength selected is the first available or the least loaded. Notice that the Least Loaded algorithm becomes the First Fit for monofibre networks.

A. Blocking Probability versus size of the WRs

Simulations have been carried out on the network topology shown in Figure 13 that consists of 9 nodes, where 2 of them are source nodes and other 2 are destination nodes.



Route 1-4A: OXC1-OXC2-OXC3-OXC4
Route 1-4B: OXC1-OXC7-OXC8-OXC4
Route 9-4A: OXC9-OXC8-OXC7-OXC4
Route 9-4B: OXC9-OXC2-OXC3-OXC4
Route 1-6A: OXC1-OXC2-OXC5-OXC6
Route 1-6B: OXC1-OXC7-OXC8-OXC4-OXC6
Route 9-6A: OXC9-OXC2-OXC5-OXC6
Route 9-6B: OXC9-OXC2-OXC3-OXC4-OXC6

Figure 13. Topology used in preliminary evaluation.

However, unlike the illustrative example described in Section 4, in this case the number of fibres and wavelengths is variable. Call arrivals are modelled by a Poisson distribution, the connection holding time is assumed to be exponentially distributed, and each arrival connection requires a full wavelength on each link it traverses.

As mentioned in previous sections an enhancement of the *PBR* mechanism is proposed to reduce the algorithm complexity and to increase the scalability. To evaluate this proposal, the effect of varying the number of *WR*s bits in the ratio of blocking is measured. Simulations are obtained by applying the *PBR* to the topology of the Figure 13. Figure 14 and Figure 15 show the blocking probability produced when varying the number of *WR*s bits applying the *RWP-f* and the *RWP*-o algorithms on the topology of Figure 13 for different conditions, that is, 1, 2 and 4 fibres per link, 6 and 8 wavelengths per fibre and different traffic loads per each source-destination pair. From the obtained results, the optimal number of bits depends on different parameters such as the traffic load, number of wavelengths and fibres. Just as an example, in Figure 14.a) the minimum number of blocked connections for the *RWP-f* algorithm, with 6 lambdas, 1 fibre and 2 Erlangs is produced for 9 bits of *WR*. Note that the number of entries of the *PT* depends on the number of bits of the corresponding *WR*; if the number of bits is *n* the number of entries of the *PT* will be 2^n. We can conclude, after analyzing the results in Figure 14, that in terms of performance having 0 bits the *WRs* is good enough, and even in most cases presents the best behaviour. With this simplification of the algorithm, the *PT*s are only of one entry (i.e., only one two-bit counter per route and wavelength).

On the other hand comparing the results for the two options used to check the *PT*s (remember that the *RWP-f* checks in a fixed order, and *RWP-o* checks depending on the wavelength availability from the point of view of the source node), the results are in almost all the cases better for the *RWP-o* than for the *RWP-f* algorithm. In Figure 14.d) we can see an exception, the *RWP-f* algorithm for 6 lambdas, 2 fibres and 5 Erlangs performs better than the *RWP-o*. Due to the reasons exposed, from now only results for the *RWP-o* algorithm without *WR*s are presented in the next subsections. All the results presented in this subsection are the mean among five simulations with a 95 % level of confidence.

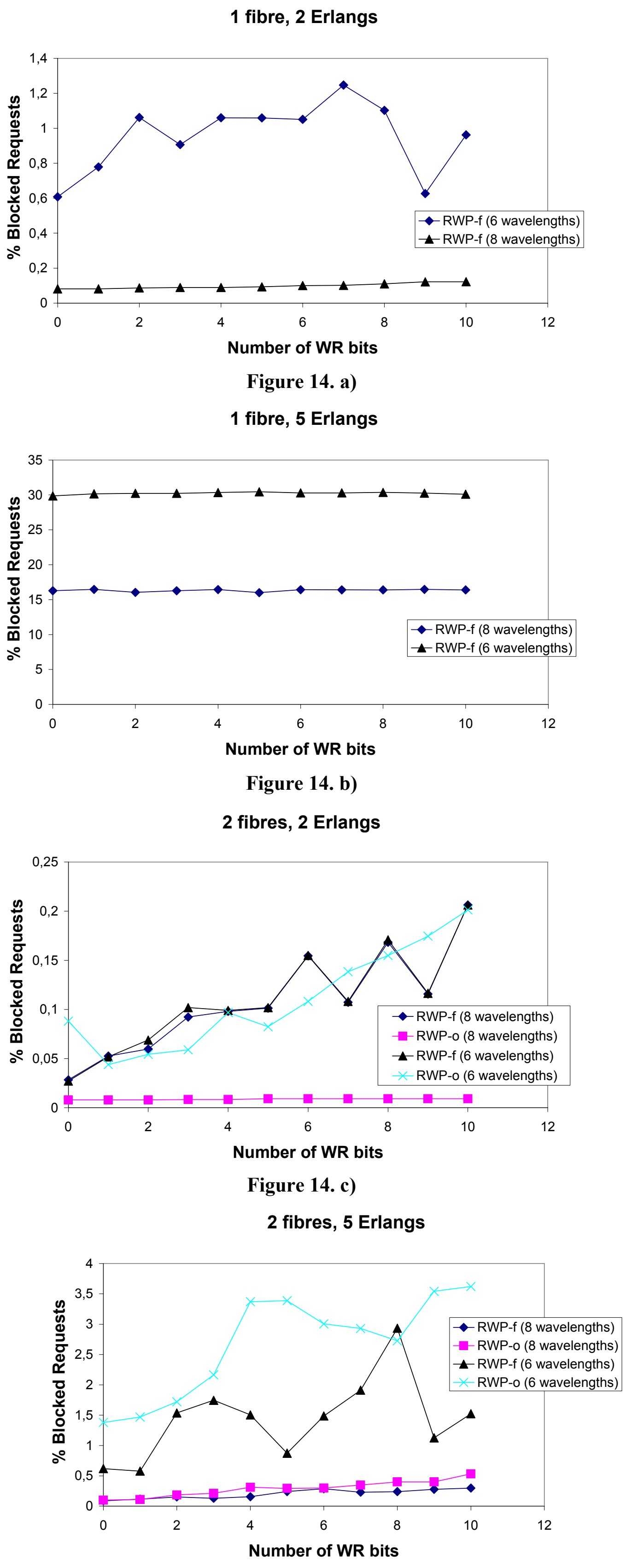


Figure 14. a)

Figure 14. b)

Figure 14. c)

Figure 14. d)

Figure 14. Percentage of blocked connection versus number of *WR* bits for *RWP*-f and *RWP*-o algorithms (1 and 2 fibres).

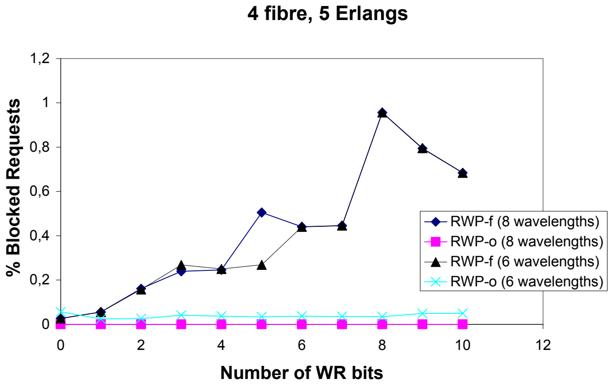


Figure 15. a)

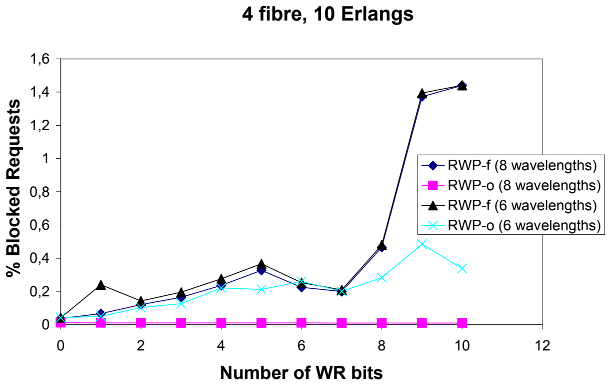


Figure 15. b)

Figure 15. Percentage of blocked connection versus number of *WR* bits for *RWP*-f and *RWP*-o algorithms (4 fibres).

B. Blocking Probability versus Traffic Load

A set of simulations have been carried out on the topology of Figure 13, varying the time between the updating from 1 to 50 units of time, and the results are presented in Figure 16 (1 and 2 fibres, for 2 and 5 Erlangs) and Figure 17 (4 fibres for 5 and 10 Erlangs). In Figure 16 only results for 2 and 5 Erlangs are presented since the percentage of blocked connections for 10 Erlangs is very high for both algorithms. On the other hand, in Figure 17 (4 fibres) results for 5 and 10 Erlangs are represented since blocking is 0 for 2 Erlangs for both algorithms and for the range of updating values, the number of blocked connections is 0. Notice that in Figure 16 and Figure 17 the *RWP-o* algorithm does not vary with the time between updating because it does not need network state update messages.

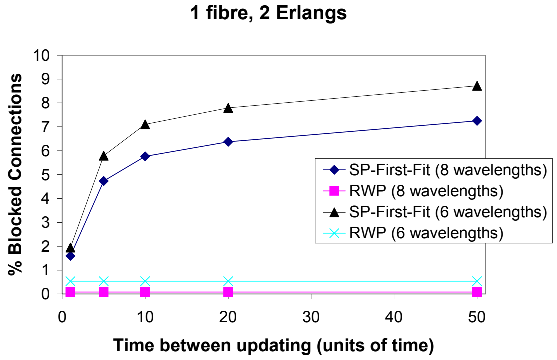


Figure 16. a)

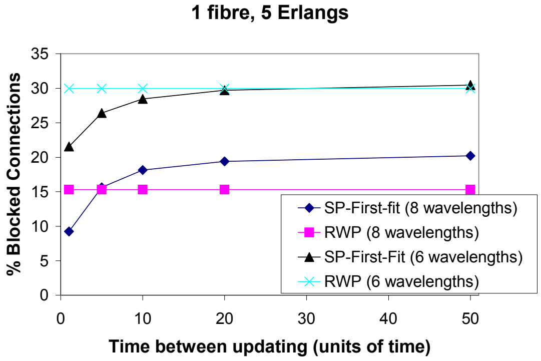


Figure 16. b)

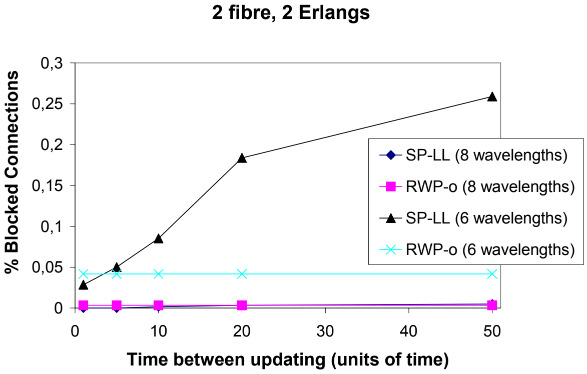


Figure 16. c)

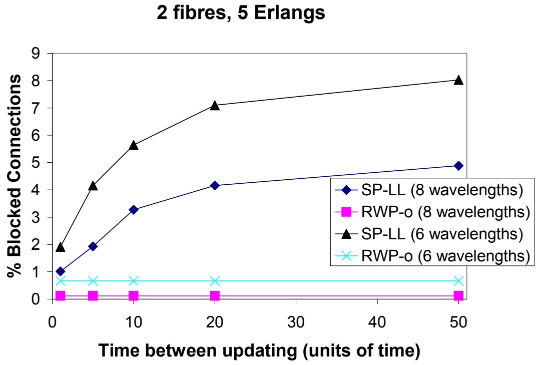


Figure 16. d)

Figure 16. *RWP* versus *SP-First-Fit* (1 fibre) and versus *SP-LL* (2 fibres).

In Figure 16.a) the results obtained for 1 fibres, 2 Erlangs and 6 or 8 lambdas depict that the *RWP* algorithm outperforms the *SP*-First-Fit algorithm, even when the update messages are flooded every unit of time. For 5 Erlangs (Figure 16.b)) and 8 lambdas the *RWP* algorithm obtains similar results than the *SP*-First-Fit algorithm with updating every 5 units of time. But for 6 lambdas and 5 Erlangs, the *RWP* algorithm only performs similar to the *SP*-First-Fit with updating every 20 units of time. Notice that in this case the percentage of blocked connections for both algorithms is high because with 6 lambdas, 1 fibre and 5 Erlangs the network is overloaded.

Results for 2 fibres are shown in Figure 16.c) and Figure.16.d). For 2 Erlangs and 8 lambdas both algorithms, *RWP-o* and *SP-LL* (Shortest Path- Least Loaded), have a blocking percentage practically equal to 0. However, for 6 lambdas the *RWP-o* algorithm has similar performance than the *SP-LL* with updating every 5 units of time. On the other hand, for 5 Erlangs (Fig.16.d)) the *RWP-o* algorithm outperforms the *SP-LL* algorithm even updating every unit of time.

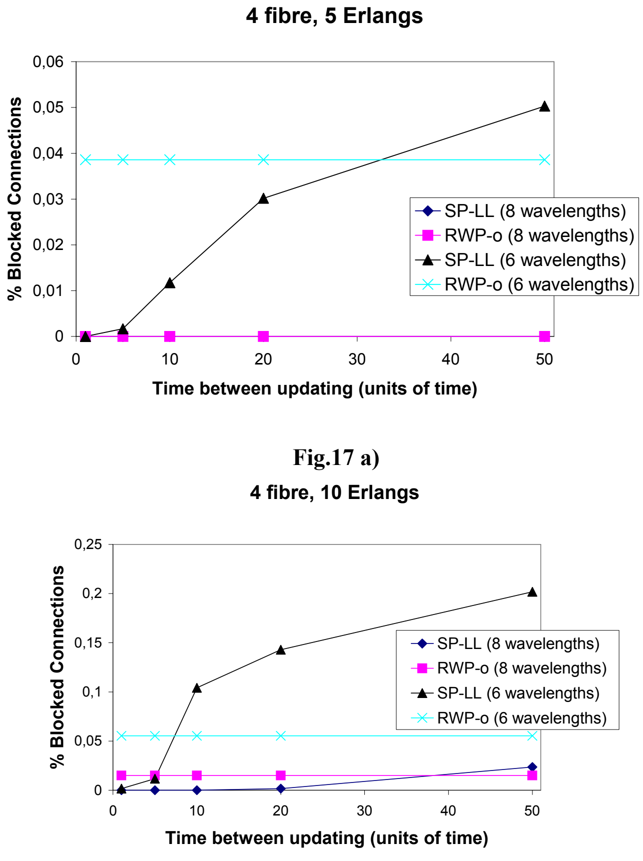


Fig.17 a)

Fig. 17 b)

Figure 17. *RWP-o* versus *SP_LL* for 4 fibres.

Results in Figure 17 correspond to simulations carried out with 4 fibres. For 5 Erlangs (Fig.17.a)) and 8 lambdas both algorithms have practically 0% of blocked connections. Instead, for 6 lambdas the range of the percentage of blocking is very close to zero, between 0% and 0,05%, and the *RWP-o* algorithm has similar results than the *SP-LL* with updating between 20 and 50 units of time.

For 10 Erlangs (Fig.17.b)) and 8 lambdas the *RWP-o* algorithm results crosses the results of the *SP-LL* algorithm when the updating is between 20 and 50 units of time. It is also observable that for 6 lambdas the *RWP-o* algorithm crosses the results of the *SP-LL* algorithm when the updating is between 5 and 10 units of time.

Summarizing, the *RWP-o* algorithm outperforms the *SP-LL* algorithm or has similar results when the updating is every 5 units of time and the parameters of traffic (traffic load, number of wavelengths and fibres) are medium (blocking between 0,5% and 20%). But if the network is overloaded (see Fig.16.b)) the *SP-LL* has better performance. On the other hand when the network is underloaded and the results of blocking are very close to zero, in some cases the *SP-LL* also outperforms the *RWP-o* algorithm (see Fig.17.b)). In this case the differences between both algorithms are negligible. The results of the *PBR* mechanism show that the routing based on prediction is a valid option because of both its capability of learning how to assign routes and the significant signalling overhead reduction.

C. Comparison of Route and Wavelength Usage.

The observation from previous results of performance in terms of blocking probability is that the algorithms based on the *PBR* mechanism deliver in the better way the traffic requests between the different routes and wavelengths. It is possible to think that this beneficial effect could be because the *PBR* mechanism assigns the routes and wavelengths in a random manner. To check this possibility in the next set of simulations it is compared how the different algorithms deliver the requests between the different routes and wavelengths. The algorithms compared are the *RWP* based on the *PBR* mechanism, the Shortest-Path algorithm combined with the First-Fit; and the Shortest-Path combined with a random wavelength assignation. This random wavelength assignation is named First-Fit (Random) because it randomly selects a wavelength among the feasible available wavelengths. The difference with the First-Fit is that the First-Fit algorithm always starts looking for a available wavelength of less index; and the First-Fit (Random) starts looking

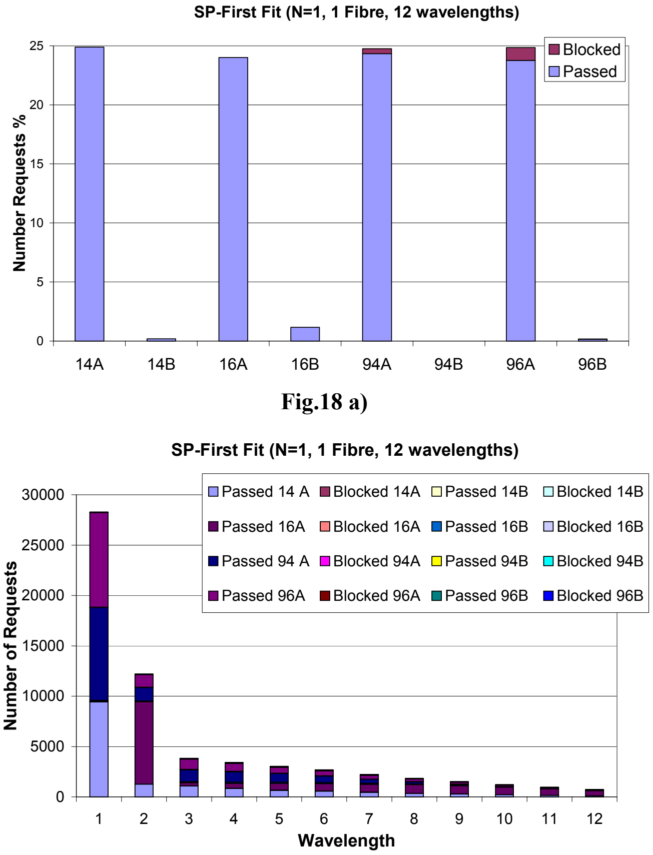


Fig.18 a)

Fig.18 b)

Figure 18. Path and Wavelength Assignment for the *SP-First Fit* algorithm.

for a randomly selected index the available wavelengths. The set of simulations have been carried out on the topology of Figure 13 for a configuration of 12 wavelengths (lambdas) per fibre and 1 fibre per link; and 1 Erlang of traffic load. Remember that for only 1 fibre the Least Loaded becomes the First-Fit. The update of network information for *SP*-First-Fit and *SP*-First-Fit(random) is every unit of time. The results of percentage in blocked connections for this configuration are 0,45% for *RWP*, 1,53% for *SP*-First-Fit and 2,31% for *SP*-First-Fit(random). Figure 18 a) represents how the connection requests are delivered by the *SP*-First-Fit among the 2 possible paths for every source destination pair. Figure 18 b) shows how the connection requests are delivered by *SP*-First-Fit among the 12 possible wavelengths. It is observable that First-Fit selects preferably wavelengths with fewer indexes and the first shortest path. Figure 19 a) and 19 b) show the same results for the *SP* algorithm combined with a First-Fit (random) wavelength assignment. In this case the requests are delivered proportionally among all the wavelengths. And finally, in Figure 20 a) and b) it is showed how the *RWP* algorithm delivers the connection requests among

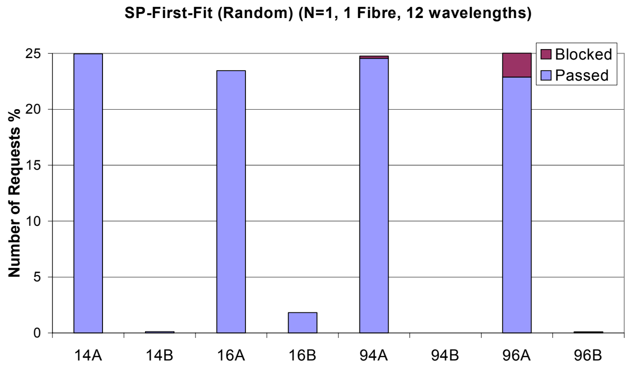


Fig.19 a)

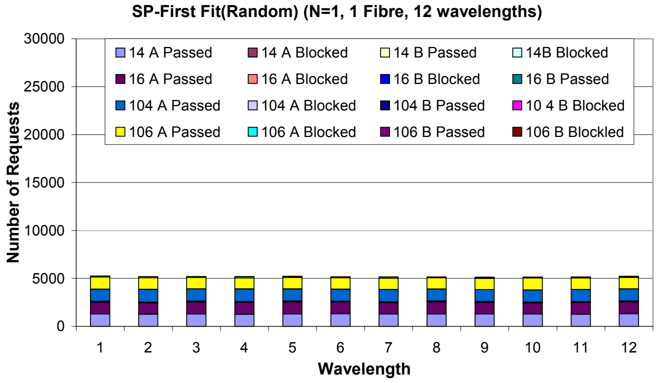


Fig.19 b)

Figure 19. Path and Wavelength Assignment for the *SP-First Fit*(Random) algorithm.

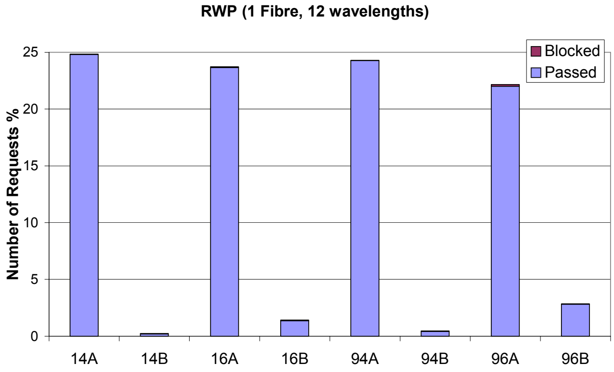


Fig.20 a)

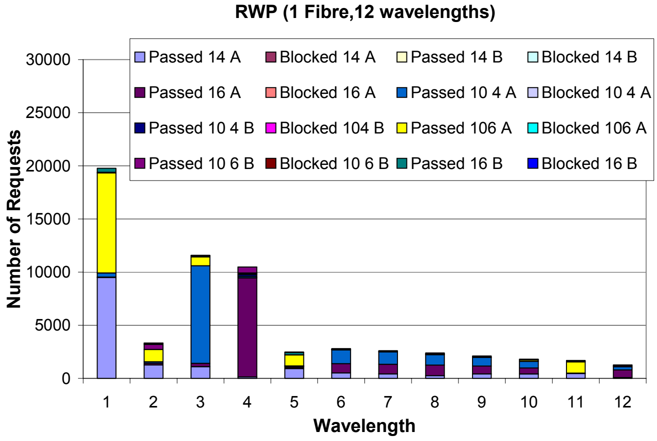


Fig.20 b)

Figure 20. Path and Wavelength Assignment for the *RWP* algorithm.

the 2 possible paths and among all the wavelengths respectively. The algorithm based on the *PBR* mechanism does not assign the wavelengths randomly; there is a pattern different from the First-Fit and the First-Fit (random) pattern. In addition, the *PBR* mechanism selects lightly more times the alternative path than the other two algorithms.

7.5.2. Results in the PanEuropean Network.

A set of simulations have been carried out on the topology of the PanEuropean network shown in Figure 21. The simulation environment consists of the following features: 2 fibres per link; and 8 wavelengths per fibre. In the first set of simulations the nodes Madrid, Frankfurt, Stockholm and Dublin act as source nodes and destinations nodes. This means 12 source-destination node pairs. A Poisson distribution models connection arrival on the wavelength switching network. The *RWP-o* algorithm is compared with the *SP-LL* (Shortest Path combined with Least Loaded) when *SP-LL* has an ideal updating (that is, it has always all the network state information) and also when the network state information is updated every 1, 5 or 10 units of time. Results in percentage of blocked connections are presented for 0,1, 0,2, 0,5, 1 and 5 Erlangs of traffic load between every source-destination node pair. All the traffic loads simulated have 10 units of time of holding time and the corresponding inter-arrival time is adjusted to achieve 0,1, 0,2, 0,5, 1 and 5 Erlangs. For

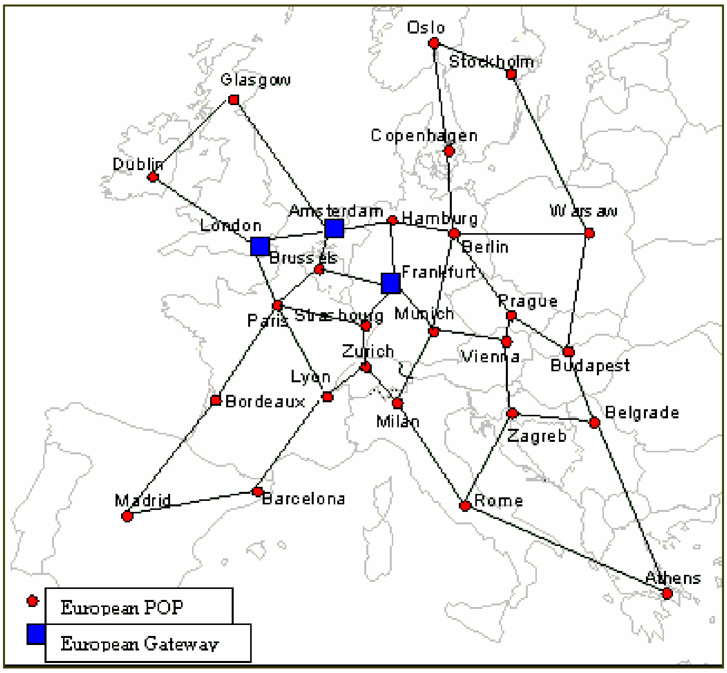


Figure 21. PanEuropean Network topology.

example for 0,1 Erlangs the holding time is 10 and the inter-arrival time is 100; or for 5 Erlangs the holding time is 10 and the inter-arrival time is 2. Results for the *SP-LL* algorithm are presented for ideal updating and for updating every 1, 5 and 10 units of time. This means for example that for updating every unit of time, during the average holding time, 10 units of time, there are in mean 10 update messages; or if updating is every 5 units of time, in mean during the holding time, 2 update messages are flooded through the network. Ideal updating is physically impossible, even updating every unit of time is physically unaffordable because every unit of time all the source nodes would have the same updated network state information. A discussion about which is the possible update of network state information is presented in the next subsection.

In Figure 22 there are represented the results of percentage of blocked connections versus the traffic load for the *RWP-o* and the *SP-LL* algorithms for 8 wavelengths per fibre and 2 fibres per link. Both algorithms select a route among all the possible routes of the network topology. That is, *SP-LL* selects the shortest lightpath (route and wavelength), and if there are more than one shortest route it selects that with more available wavelengths of that colour. The *RWP-o* algorithm selects the shortest lightpath with two-bit counter lower than 2 and output link availability, and if there are more than one route it selects that with more available wavelengths of that colour, but using local information. The first observation from Figure 22 is that for traffic loads from 0,1 to 1 Erlangs the network has enough

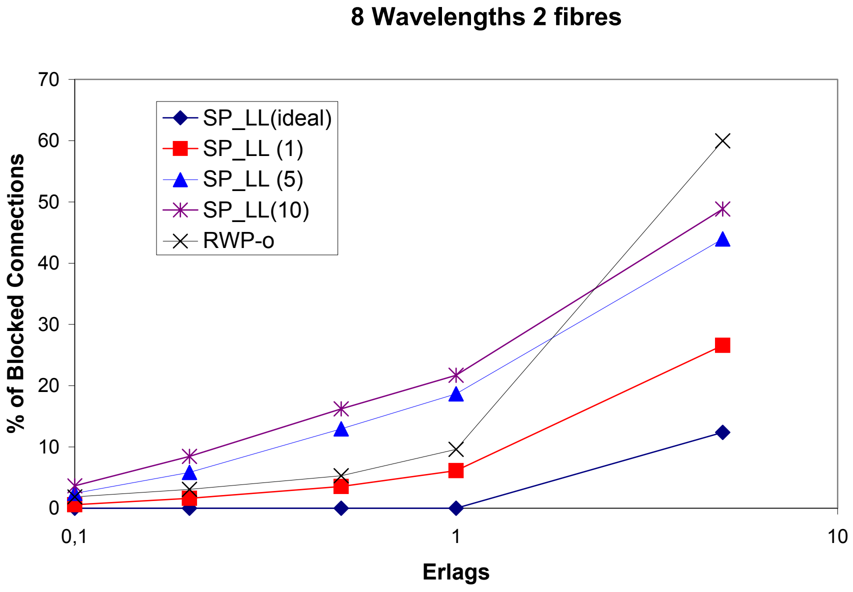


Figure 22. *SP-LL* versus *RWP-o*.

resources to set up all the connection requests. The *SP-LL* with ideal updating produces 0% of blocked connections from 0.1 to 1 Erlangs. This means that the network has enough resources to cope with the traffic load. But, when the time between updating increases the blocked connections increase too. Note that when updating every unit of time the inaccuracy of the network state information is not avoided. This is produced when two different nodes select lightpath at the same unit of time, but one sets up the connection before the other; and then, the second utilizes out-of-date information. From 0,1 to 1 Erlangs only *SP-LL*(ideal) and *SP-LL*(1) outperforms the *RWP-o* algorithm; but *RWP-o* outperforms *SP-LL*(5) and *SP-LL*(10). On the other hand for high traffic load, 5 Erlangs, *RWP-o* has the worst results.

In the next set of simulations it is compared the performance of both algorithms when the number of possible routes to select by the algorithm is reduced. Figure 23.a) shows the percentage of blocked connections obtained by the *SP-LL* algorithm considering ideal updating when routes selected are either all the possible routes or the 2 shortest routes or the 2 shortest and link disjoint routes. Figures 23.b), 23.c), 23.d) shows the results of the *SP-LL* algorithm considering updating every 1, 5 and 10 units of time when selected routes are either all the routes or the two shortest, or the 2 shortest and link disjoint routes. Based on the obtained results we can conclude is that only when the updating is ideal is useful to select among all the routes (Figure 23.a). Instead, when there is certain inaccuracy, updating every 1, 5, or 10 units of time, the *SP-LL* algorithm presents the best results when selecting among the 2 shortest and link disjoint routes. This means that larger number of routes does not mean better performance. On the other hand, in Figure 23.e) we observe that the reduction obtained by the *RWP-o* algorithm in the blocking ratio when selecting the route between the two shortest and link disjoint routes is higher than that obtained by the *SP-LL* algorithm in the same context. This is due to the fact that the lower the number of routes the lower the number of two-bit counter to train. With 2 shortest and link disjoint routes it has to train 8(wavelengths-) x 2(routes) = 16 two bit-counters per source destination node pair. However, if the algorithm could select among all the possible routes the number of two-bit counters will be 8(wavelengths) x Number of Possible routes between source destination nodes. Note that the two-bit counters are trained by means of the produced blocked connections.

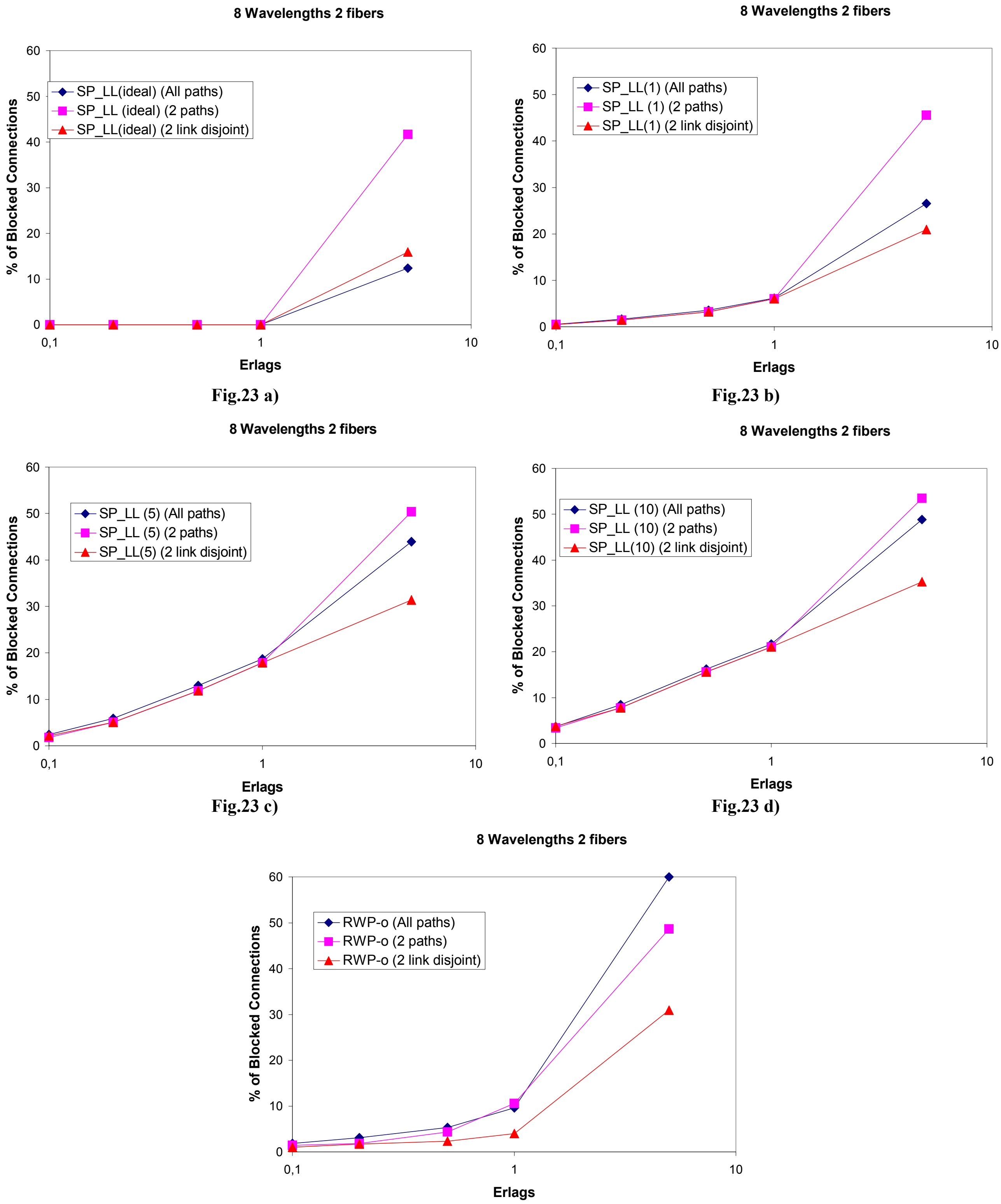


Fig.23 a)

Fig.23 b)

Fig.23 c)

Fig.23 d)

Fig.23 e)

Figure 23. Effect on blocking performance of the number of possible routes.

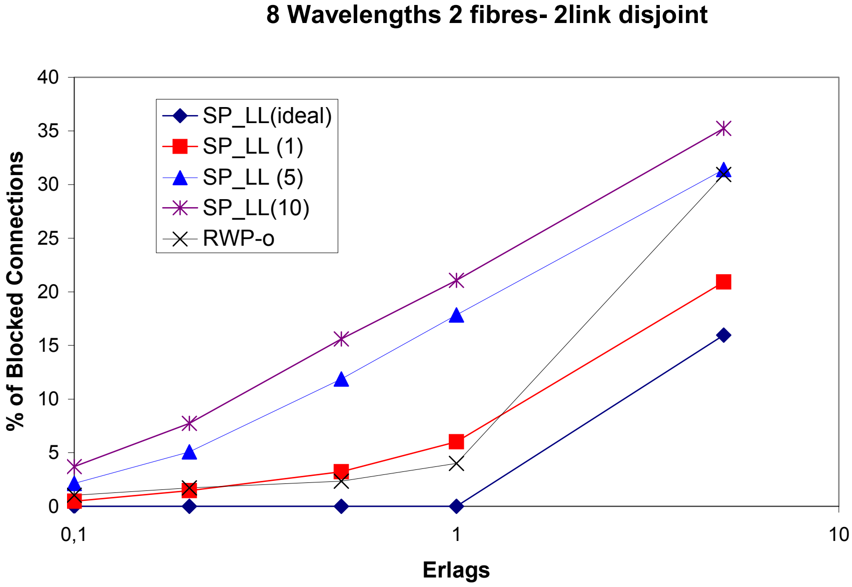


Figure 24. *RWP-o* versus *SP-LL* with 2 link disjoint routes.

Figure 24 shows the percentage of blocked connections produced by the *RWP-o* algorithm with 2 shortest and link disjoint routes, compared with the results obtained by the *SP-LL* also with 2 shortest and link disjoint routes. This graphic shows the improvement of the *RWP-o* algorithm with 2 link disjoint routes, because it outperforms the *SP-LL* even with updating every unit of time from 0,1 to 1 Erlang. In this range only the ideal case of *SP-LL* outperforms the *RWP-o* algorithm. For high traffic load the *RWP-o* has the same performance as the *SP-LL* with updating every 5 units of time. Notice that the load in Erlangs represents the load between every source destination pair of nodes. Just as an example, 1 Erlang means that there is 1 Erlang load between every one of the 12 source destination node pairs.

In the previous simulations it is assumed that only 4 nodes in PanEuropean network act as source and destination. This means that 12 possible connections between source and destination nodes can be established. In the next set of simulations 10 of the 28 nodes of the PanEuropean network will act as source and destination nodes. These nodes are: Madrid, Barcelona, Paris, Dublin, Milan, Frankfurt, Amsterdam, Prague, Stockholm and Athens. In this case 90 possible connections between source and destination nodes can be established. The objective is to check if the previous results with 12 possible connections can be extrapolated when more nodes act as source and destination; and if the *PBR* mechanism shows the same behaviour. Results are shown in Figure 25 and Figure 26. Both algorithms

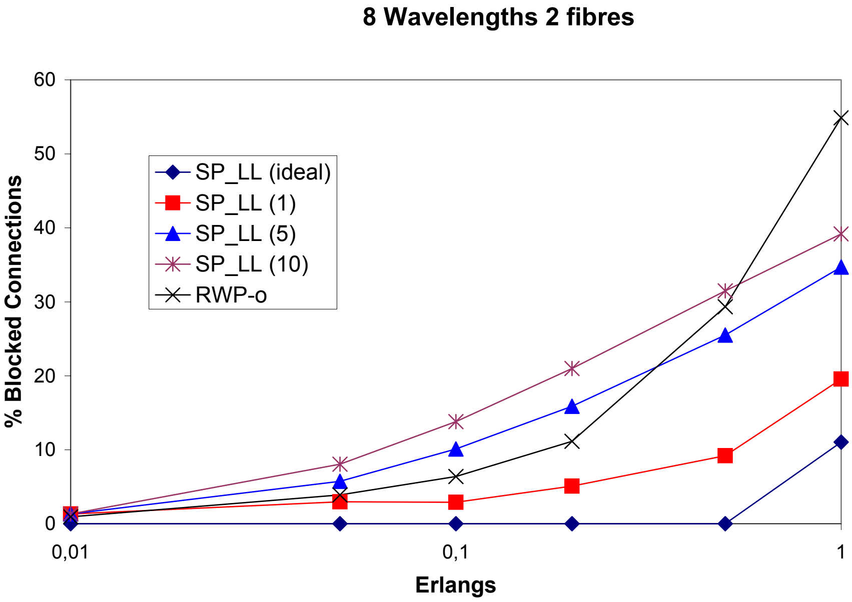


Figure 25. *SP-LL* versus *RWP-o*.

select among all the possible routes between every source destination pair of nodes in Figure 25; and in Figure 26 both algorithms (*RWP-o* and *SP-LL*) can only select between the two shortest and link disjoint routes. Figures 25 and 26 show a similar behaviour for 10 source nodes than Figure 22 and Figure 24 for 4 source nodes. For low and medium traffic (from 0,01 to 0,2 Erlangs) the *RWP-o* algorithm outperforms the *SP-LL*(5) algorithm. Only the *SP-LL* with ideal updating and updating every unit of time presents better performance than the *RWP-o*. Note that in Figures 22, 23, 24, 25 and 26 the x-axis refers to the traffic load in Erlangs between every source destination pair of nodes. With 4 nodes acting as source and destination, there are 12 source-destination combinations. However, with 10 nodes acting as source and destination there are 90 source-destination combinations. For high traffic load, 0,5 to 1 Erlang, and when selecting among all the possible routes (Figure 25) the *RWP-o* presents worse performance than the *SP-LL* with updating every 10 units of time. But when both algorithms can select only between 2 links disjoint routes the *RWP-o* outperforms the *SP-LL*(5) (5 Erlangs) or has similar results (1 Erlang). On the other hand, it is possible to confirm that the impact in the blocking ration of selecting the route between the 2 shortest and link disjoint is higher in the *RWP-o* than in the *SP-LL* algorithm.

The *RWP-o* algorithm degrades its performance more rapidly for high traffic load than the *SP-LL*. The main reason is the two-bit counter inertia for changing the lightpath selection.

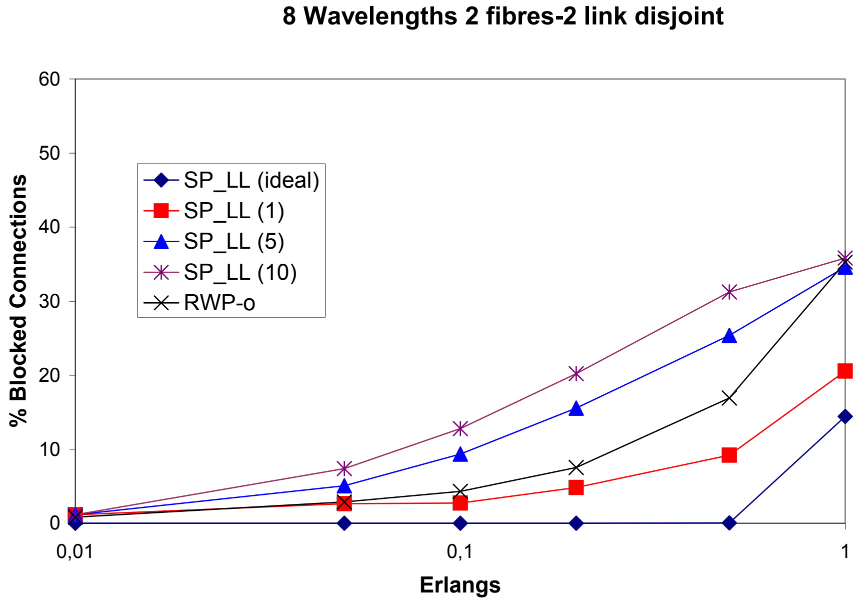


Figure 26. *RWP-o* versus *SP-LL* with 2 link disjoint routes.

High traffic load means that in mean more connections are requested per unit of time. In this scenario the two-bit counters are too slow to cope with the traffic pattern. This behaviour is lightly mitigated when the *RWP-o* algorithm can only select among 2 link disjoint routes. When there are more routes to select, on the one hand it might not be beneficial because longer routes are selected wasting more network resources and not avoiding to establish later connection requests. Only when the *SP-LL* has all the updated network information, *SP-LL* (ideal), is beneficial to select among all the possible routes. But when there is a certain degree of inaccuracy it is preferable to select only among 2 link disjoint routes. On the other hand, when the *RWP-o* can select among more routes it has to train more two-bit counters. If there is high traffic load it is more probable that the *RWP-o* algorithm tries to select more routes than if the traffic is light. Just as an example, if there are only 2 possible routes and with high traffic load, if these 2 routes cannot be selected (because either the two-bit counters are greater than 2 or e there is not output link availability), the *RWP*-o algorithm would not select any route. But with more routes, the *RWP-o* algorithm would select next routes. In this scenario the probability that the *RWP-o* algorithm selects a specific route is lower when there are more routes. The two-bit counters of the lightpaths are trained or learn by means of the blocked connections. When the time from a lightpath is selected to the time it is selected again is long the two-bit counters cannot cope the pattern of behaviour of the traffic.

7.5.3. Stabilizing time

In this section it is computed an approximation to the stabilizing time in the PanEuropean Network. The main objective is to compute the maximum updating frequency, i.e. the minimum updating interval, matching the physical constraints. The stabilizing time is the time required for nodes to update network state information [63]. The computation of the stabilizing time is done with the same assumptions as done by Zang et al in [63]. It is assumed that the signalling messages with update information are delivered in a packet-switched control network. This control network is implemented on an out-of-band supervisory channel that operates on its own wavelength. For this reason the signalling overhead due to the update messages would not be a problem for the *SP-LL* algorithm. The control layer has the same topology as the physical network; and all packets are routed by shortest paths. It is also assumed that the signalling (update) messages are routed via the path with the shortest propagation delay in the control network.

In [63] authors utilize a holding time of 100 ms. This value corresponds to a very dynamic traffic. The exact value of the holding time is out of the scope of this Thesis, but only for high dynamic traffic the routing inaccuracy problem due to the propagation delay comes up. For this reason it will be assumed a holding time of 100 ms to estimate the stabilizing time in the PanEuropean Network. In the previous simulations of percentage of blocked connections in the PanEuropean Network a holding time of 10 units of time has been used. For the next computations, it will be assumed that 100 ms corresponds to 10 units of time. That is, 1 unit of time is 10 ms.

Zang et al in [63] compute the stabilizing time assuming that all nodes send an update message to all other nodes. Then, they compute the stabilizing time of a single node as the time that an update message needs to reach the farthest node. They assume that the time needed to reach to the farthest node is only due to delay considerations. Hence, no time to transmit or switch the control packets is considered. It is known that the light propagation delay over fibre is 5 μs/km (0,005 ms/km).

In the first simulations in PanEuropean Network, nodes Madrid, Frankfurt, Stockholm and Dublin are considered source and destination nodes (Figures 22, 23 and 24). These nodes have to send update message to all the other source nodes. First, it is computed for each source node the maximum time needed to send update message to all the other source nodes. Tables 1, 2, 3 and 4 show these delay times taking into account the distance in kilometres between different nodes. Distance in kilometres is extracted from [64].

Table 1. Propagation delay in ms from Madrid.

Route	**Shortest Route**	**km**	**Delay (ms)**
Madrid-Frankfurt	Madrid-Bordeaux-Paris-Strasbourg-Frankfurt	2452	2452 x 0,005= 12,26
Madrid-Stockholm	Madrid-Bourdeaux-Paris-Strasbourg-Frakfurt-Hamburg-Berlin-Warsaw-Stockholm	5310	5310 x 0,005 = 26,55
Madrid-Dublin	Madrid-Bordeaux- Paris- London-Dublin	2785	2785 x 0,005 = 13,92
Maximum			26,55

Table 2. Propagation delay in ms from Frankfurt.

Route	**Shortest Route**	**km**	**Delay (ms)**
Frankfurt-Madrid	Frankfurt- Strasbourg- Paris- Bordeaux-Madrid	2452	2452 x 0,005 = 12,26
Frankfurt-Stockholm	Frankfurt-Hamburg-Berlin-Warsaw-Stockholm	2858	2858 x 0,005 = 14,29
Frankfurt-Dublin	Frankfurt-Strasbourg-Paris-London-Dublin	2075	2075 x 0,005 = 10,38
Maximum			14,29

Table 3. Propagation delay in ms from Stockholm.

Route	**Shortest Route**	**km**	**Delay (ms)**
Stockholm-Madrid	Stockholm-Warsaw-Berlin-Hamburg-Frankfurt-Strasbourg-Paris-Bordeaux-Madrid	5310	5310 x 0,005 = 26,55

Stockholm-Frankfurt	Stockholm-Warsaw-Berlin-Hamburg-Frankfurt	2858	2858 x 0,005 = 14,29
Stockholm-Dublin	Stockholm-Warsaw-Berlin-Hamburg-Amsterdam-London-Dublin	4048	4048 x 0,005 = 20,24
Maximum			26,55

Table 4. Propagation delay in ms from Dublin.

Route	**Shortest Route**	**km**	**Delay (ms)**
Dublin-Madrid	Dublin-London-Paris-Bordeaux-Madrid	2785	2785 x 0,005 = 13,93
Dublin-Frankfurt	Dublin-London-Paris-Strasbourg-Frankfurt	2075	2075 x 0,005 = 10,38
Dublin-Stockholm	Dublin-London-Amsterdam-Hamburg-Berlin-Warsaw-Stockholm	4048	4048 x 0,005 = 20,24
Maximum			20,24

The maximum time obtained from Tables 1,2,3 and 4 is 26,55 ms. That is, 26,55 ms is the minimum time that updating the network state is physically possible. Note that in this computation it is considered that update messages are sent in an out-of-band control network, without wasting resources (wavelengths) of the data network. For this reason signalling overhead produced by these update message is not taken into account. The stabilizing time of 26,55 ms means that the results of percentage of blocked connections obtained for *SP-LL* with ideal updating and updating every 1 unit of time (10 ms) are physically unaffordable. Then, only comparison between the *RWP* and the *SP-LL* algorithms for updating from 5 units of time is valid. And in this case, the algorithm inferred from the *PBR* mechanism outperforms the *SP-LL* algorithm from 0,1 to 1 Erlang. Only for high traffic load, 5 Erlangs, the *PBR* degrades its performance. But for 5 Erlangs the network does not have enough resources as shown by the 12,37% of blocked connections obtained by the *SP-LL* algorithm with ideal updating.

In the second set of simulations in PanEuropean Network, Figures 25 and 26, there are 10 nodes acting as source and destination. Taking into account that the propagation delay is determined by the farthest nodes, these nodes are also Madrid Stockholm with a propagation delay of 26,55 ms. For this reason only results for *SP-LL* (5), *SP-LL*(10) and *SP-LL*(20) are physically affordable, and they can be compared with *RWP-o* results.

8. The Prediction-Based Routing Mechanism for Hierarchical WDM Networks

8.1. Introduction to Hierarchical *WDM* Networks.

In this section it is presented a hierarchical routing overview to introduce the benefits of applying the Prediction-Based Routing Mechanism to hierarchical networks.

A hierarchical network architecture comes out as one of the hard recommendations stated at the *ASON* specifications [27] to guarantee network scalability. A whole hierarchical network structure should be subdivided into routing areas (*RAs*), (see Figure 27 as an example) containing physical nodes with similar features. The *RA* nodes should exchange topology and resource information among themselves in order to maintain an identical view of the *RA*. Each *RA* should be represented by a "Logical Routing Area (*LRA*) Node" in the next hierarchical level. The required functions to perform this role should be executed by a node called the "Routing Area Leader" (*RAL*). This node will receive complete topology state information from all *RA* nodes and will send information up to the *LRA* node. The propagated information only includes the information needed by the higher level.

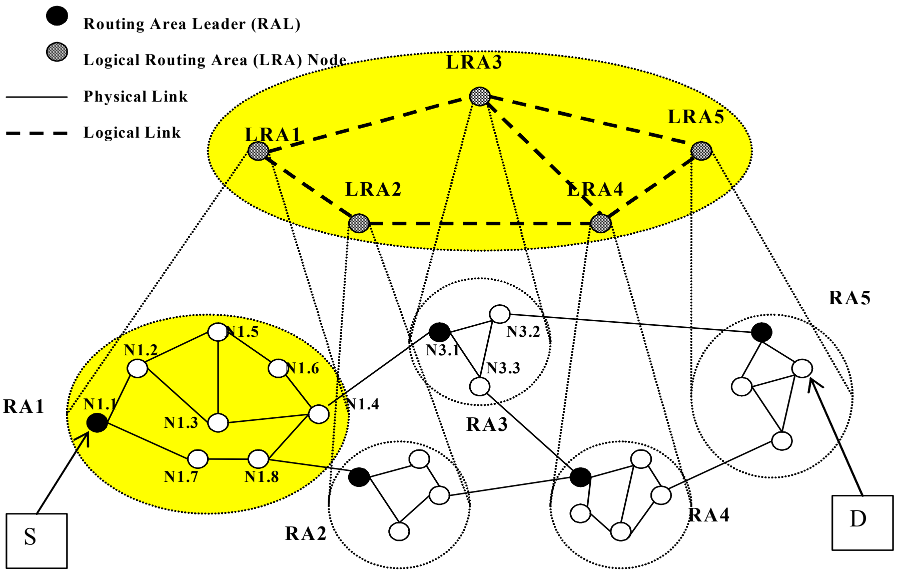


Figure 27. A hierarchical network structure.

The main advantage of hierarchical routing is to reduce large signalling overhead while providing efficient routing. Therefore to achieve this goal, traditional flat network structures must be properly modified to fulfil that *ASON* recommendation. Main concepts to be modified are those related to signalling and routing, such as the network information aggregation, the network information dissemination, the updating policies and the routing algorithm.

A. Aggregation Scheme

As stated above the *RAL* receives complete topology state information from all the network nodes in its hierarchical level. This information is aggregated before being forwarded to the *LRA* node. The policy used to define how and which information is aggregated, is defined by some aggregation scheme.

The main benefit introduced because of using any aggregation scheme is the reduction of the amount of information to be distributed throughout the network. However, a collateral and negative effect of such aggregation scheme is that the information used to compute routes is non-complete, that is, aggregated information does not contain full information about physical links and nodes. This negative effect of the aggregation schemes contributes to increase the inaccuracy of the network state information, that is, the routing inaccuracy problem. The aggregation process will aggregate the information of several network parameters. The following network parameters were proposed for optical networks:

- *D*: Propagation delay in a link which is proportional to the fibre distance between two nodes.
- As_p: Number of available wavelength of each colour in a link

The rest of document assumes for hierarchical networks the aggregation scheme named *NAS* (Node Aggregation Scheme), which was proposed by Sánchez in [65] and [66].

B. Update policy

In traditional *RWA* algorithms the update policies are required to guarantee that the information contained in the network state databases perfectly represents a current picture of the network in order to guarantee an optimal path selection. In general update messages may be triggered by either a periodical refresh (i.e., time-based triggers) or a network change (i.e., threshold-based triggers). While the former does not take into account the network dynamics the latter can drive to a significant signalling overhead in dynamic networks i.e., networks where many new connection setups and releases occur in a short period of time. Thus, new update policies must be developed to reduce this signalling overhead while guaranteeing accurate routing information. However, there is a trade-off between the amount of update messages and the accuracy of the network state information. In fact, the larger the amount of update messages (signalling overhead) the lower the inaccuracy. Since keeping an up-to-date picture of the network is currently not affordable, a certain degree of inaccuracy will always be introduced by any update policy included in the routing protocol.

C. Routing Algorithm

ASON specifications do not recommend a routing algorithm in order to compute routing paths. However, it defines a set of features that have to be supported by any routing algorithm running over the optical networks. One of them recommends path computation based on source routing. The routing decisions are taken on the source nodes based on the global network state information contained in their network state databases. As mentioned above, several causes strongly impact on the network state information accuracy. Unlike traditional flat networks where the inaccuracy is basically introduced by the update policy in hierarchical networks such inaccuracy is introduced not only by the update policy but also by the aggregation scheme used to select the information to be disseminated around the network.

8.2. Description and Data Structures.

After this hierarchical network overview, the main advantages of introducing the Prediction-Based Routing (*PBR*) concept in hierarchical networks can be inferred. As stated above in a hierarchical network scenario the inaccuracy of the network state information is greater than in flat networks. For this reason it can be appropriate routing algorithms that do not use this out-of-date network state information, such as algorithms inferred from the *PBR* mechanism. In the next subsection there are thoroughly described two hierarchical routing algorithms based on prediction. These hierarchical prediction-based algorithms compute the route in a hierarchical structure, that is, if the destination node belongs to the same *RA* the route is completely defined. Otherwise, if the destination node belongs to a different *RA*, the route is specified by both the route from the source node to the last node on its *RA* and the different *RAs* to reach the destination node.

Concerning to the data structures, as in flat networks, in the source nodes there will be a *WR* register and a *PT* table for every wavelength on a route for every destination, but taking into account that the route will be defined in hierarchical mode.

A. Wavelengths Registers, Prediction Table and Database Table

In every source node there is a Database Network State Table containing the information of availability of all the internal links of the *RA*. This database is not updated by means of update messages in the first of the proposed algorithms; it is updated only by means of local information. However in the second proposed algorithm, the network state information (database) is updated depending on the frequency of updating. The parameter *N* represents this updating frequency. When a source node produces *N* changes, *N* lightpaths are set up or torn down; it sends an update message to the other source nodes with updated information. On the other hand, in every source node there is one *WR* and one *PT* for every route and wavelength for every possible destination, but in this case the source and destination nodes are nodes in the following hierarchical level. For example, from the Figure 27, a possible route between RA1 and RA5 is RA1-RA3-RA5, and in the node N1.1 of the RA1 there would be one *PT* and one *WR* for every wavelength for the route RA1-RA3-RA5.

8.3. *PHOR* algorithm description.

In this subsection it is presented the routing algorithm named *PHOR* (Prediction Hierarchical Optical Routing) [66], [67], which is based on modifying the *RWP* algorithm for flat networks to be applied to hierarchical networks. The main advantages of introducing the *PBR* concept in hierarchical networks is that neither update messages are required nor any aggregation process.

The algorithm works as follows. The k-shortest and link disjoint routes, A and B (assuming k = 2) are precomputed in the source nodes for every destination node. If such a destination node belongs to the same *RA* the path is completely defined. Otherwise, if the destination node belongs to a different *RA*, the route is specified by both the route from the source node to the last node on this *RA* and the different *RAs* for the rest of the route. For example in Figure 27 assuming k = 2, if the source node is the N1.1 and the destination node is the N5.4, the two shortest routes are N1.1-N1.2-N1.5-N1.6-RA3-RA5 (A) and N1.1-N1.7-N1.8-RA2-RA4-RA5 (B). There is one *WR* and one *PT* for every wavelength for these two routes. Assuming that being A and B link disjoint routes, A accounts for the shortest route and B is equal or longer than A.

Wavelengths on each route are weighted according to the minimum number of available fibres of every wavelength per link along the lightpath. This weight is used to order all different possibilities to setup the lightpath. It is important to note that this information is only from the point of view of the source node N1.1. This source node only knows how many wavelengths has assigned in every link but it does not know the real availability of the links because there are not update messages. The Prediction Tables (*PTs*) are checked in this computed order. The decision of which wavelength and route are chosen is done depending on the value of the counters of the *PTs* and the availability of the node's output links. In Figure 28 it is showed the core of the pseudo-code of the *PHOR* algorithm. Once the order for Route A has been computed (Routine Order(Route A)), then, the wavelengths of route A are checked (Routine Check(Route A)). If the algorithm does not choose any wavelength in route A, the route B is checked (Routine Check(Route B)). Afterwards, if wavelength and route are not assigned yet, the algorithm tries to assign the wavelength in route A only checking the availability of the node's output link towards route A (CheckF(Route A)), and if CheckF(Route A) does not assign wavelength the routine CheckF(Route B) tries to assign the wavelength in route B only checking the availability of the node's output link towards route B.

As it is done in the algorithm proposed for flat networks the two-bit counters of the *PTs* are updated in order to train them. If the connection can be established in the lightpath, route and wavelength, selected the corresponding two-bit counter is decreased. But if the connection is blocked the two-bit counter is increased.

```
1.      Order(Route A)
            ($o_0$, $o_1$……$o_{number\_of\_wavelengths -1}$ is the index wavelength order for checking Route A)
2.      Check(Route A):
            i=0;
3.          while (route is not assigned and i<number_of_wavelengths){
4.                  if (PTcounter($o_i$)<2 and wavelength $o_i$ is available in outgoing link to route A)
                            { assign route A and wavelength $o_i$;
                            if connection is established decrease PTcounter($o_i$)
                            else increase PTcounter($o_i$)
                    }endif
             i++;
            }endwhile
            endCheck
5.      If (route is not assigned) {
6.      Order(Route B)
          ($o_0$, $o_1$……$o_{number\_of\_wavelengths -1}$ is the index wavelength order for checking Route B)
7.      Check(Route B):
            i=0;
8.          while (route is not assigned and i< number_of_wavelengths){
9.                  if (PTcounter($o_i$)<2 and wavelength $o_i$ is available in outgoing link to route B )
                            { assign route B and wavelength $o_i$;
                            if connection can be established decrease PTcounter($o_i$)
                            else increase PTcounter($o_i$)
                    }endif
            i++;
          }endwhile
             endCheck
        }endif
10.     If (route is not assigned){
11.     CheckF(Route A):
            i=0;
12.         while (route is not assigned and i< number_of_wavelengths){
13.                 if (wavelength i is available in outgoing link to route A)
                            { assign route A and wavelength i;
                            if connection is established decrease PTcounter(i)
                            else increase PTcounter(i)
                     }endif
            i++;
          }endwhile
          endCheckF
14.         If (route is not assigned) {
15.         CheckF(Route B):
                i=0;
16.             while (route is not assigned and i< number_of_wavelengths){
17.                 if (wavelength i is available in outgoing link to route B)
                            { assign route B and wavelength i;
                            if connection is established decrease PTcounter(i)
                            else increase PTcounter(i)
                    }endif
                i++;
                }endwhile
            endCheckF
            }endif
        }endif
```

Figure 28. Pseudo-code of the *PHOR* algorithm.

8.4. *BAPHOR* algorithm description.

The algorithm presented in the previous subsection has some advantages and weaknesses. In order to take advantage of the benefits while reducing the weaknesses of such algorithm it is proposed a hybrid routing algorithm named *BAPHOR* (Balanced Prediction Hierarchical Optical Routing) [66][67]. This algorithm combines the benefits of a balanced based and a prediction based algorithm. The main idea underlying such algorithm is that the aggregated network state information of the external *RAs* can be replaced by a prediction about the availability through the external RAs. On the other hand, the network state information within the *RA* is flooded by an update policy and utilized by a balanced routing algorithm. Summarizing, the aggregation schemes are not necessary because the network state information is not flooded between different *RAs*. Nevertheless, updating is needed into every *RA*. Such scheme makes the dissemination process easier since dissemination is only limited to *RAs* scenarios.

The *BAPHOR* algorithm bases its decision on choosing the route and the wavelength that minimizes a hierarchical weight value, $W_h(\lambda_i)$ as it is done in the balanced routing algorithm, named *BHOR*, presented also in [66]. The *BHOR* algorithm was proposed as the hierarchical routing algorithm inferred from the ALG3 proposed in [68] and in [69] by Masip et al for flat networks. The *BHOR* algorithm calculates a hierarchical $W_h(\lambda_i)$ value by adding a weight value, $W(\lambda_i)$, of each hierarchical level. Note that the first hierarchical level is into the *RA* where the source node is. In each hierarchical level this $W(\lambda_i)$ value is $Hn\left(\frac{Od}{Cd}\right)$. These three components are the length of the selected lightpath, (*Hn*), the degree of congestion (*Cd*), and the degree of obstruction (*Od*). The length, *Hn*, is simply the number of hops. The degree of congestion, *Cd*, is the wavelength availability, that is, the minimum number of available wavelengths of that colour in that route. Unfortunately, because of the update policy the degree of congestion may not be accurate enough. For this reason, the degree of obstruction, *Od*, tries to minimize the impact of such inaccuracy on the lightpath selection process. *Od* represents the number of links on the route where such a wavelength is defined as potentially obstructed wavelength (*POW*). Assuming that the hierarchical network mechanism is based on a threshold-based updating, the *POW* definition must take into account the value of this threshold. In a threshold-based updating,

```
1.  Assign to MIN a big value and assign to MAX the value 0
    for i=0 to number_of_wavelengths - 1
2.  {       Calculate $Od(\lambda_i)$ in the internal part of route of A (into RA)
3.          Calculate $Cd(\lambda_i)$ in the internal part of route of A (into RA)
4.          $W_h A(\lambda_i) = Hn\_A\left(\frac{Od(\lambda_i)}{Cd(\lambda_i)}\right) + PTcounterA(\lambda_i)$ ;
            if(($W_hA(\lambda_i)$<MIN) OR (($W_hA(\lambda_i)$ ==MIN) and($Cd(\lambda_i)$>MAX)))
            {       ROUTE=A;
                    WAVELENGTH=$\lambda_i$;
                    MIN= $W_hA(\lambda_i)$;
                    MAX=$Cd(\lambda_i)$;
            }endif
5.          Calculate $Od(\lambda_i)$ in the internal part of route of B (into RB)
6.          Calculate $Cd(\lambda_i)$ in the internal part of route of B (into RB)
7.          $W_h B(\lambda_i) = Hn\_B\left(\frac{Od(\lambda_i)}{Cd(\lambda_i)}\right) + PTcounterB(\lambda_i)$ ;
8.          if (($W_hB(\lambda_i)$<MIN) OR (($W_hB(\lambda_i)$==MIN) and($Cd(\lambda_i)$>MAX)))
            {       ROUTE=B;
                    WAVELENGTH=$\lambda_i$;
                    MIN= $W_hB(\lambda_i)$;
                    MAX= $Cd(\lambda_i)$;
            }endif
    }endfor
9.  Assign route ROUTE and wavelength WAVELENGTH
            if connection can be established decrease PTcounterROUTE(WAVELENGTH)
            else increase PTcounterROUTE(WAVELENGTH)
```

Figure 29. Pseudo-code core of the *BAPHOR* algorithm.

network state information is updated when there are *N* changes. That is, *N* lightpaths are set up or torn down. Being *B* (any link is a bundle of *B* fibres) the total number of a certain wavelength λ_i on a link, *R* the current number of available λ_i on this link, and according to the threshold-based update policy, the wavelength λ_i is defined as *POW,* namely $\lambda^{POW}{}_i$ on a certain link, when $R \leq pr$ (being *pr* a percentage of the threshold value). Then, for every lightpath the weight calculated as $W(\lambda_i)$ stands for a balance between the number of potentially obstructed wavelengths and the real congestion. The length of the path is also included in order to avoid those paths that are either widest but too long or shortest but too narrow.

On the other hand, the *BAPHOR* algorithm, in the first hierarchical level (into the RA) computes the $W(\lambda_i)$ value as $Hn\left(\frac{Od}{Cd}\right)$. Assuming there is a *PT,* Prediction Table, for every route and wavelength in the following hierarchical level (out of the RA), the value to add for the next hierarchical level is the value of the corresponding two-bit counter. If there are more than 2 hierarchical levels, for all the levels different from the first, the value to add is a two-bit counter value. This is expressed in Eq. (5) when the number of hierarchical levels is n.

$$W_h(\lambda_i) = Hn\left(\frac{Od}{Cd}\right)(j=1) + \sum_{j=2}^{n} PTcounter(j) \qquad (5)$$

The $W(\lambda_i)$ values and *PT* counter values can be mixed in this manner because both account for more availability when they are low, and account for less availability when they are high. Assuming k-shortest paths with k = 2, for every possible source destination pair the two shortest routes, A and B are precomputed. The algorithm chooses the route, A or B, and the wavelength that minimizes the $W_h(\lambda_i)$ value, but when two $W_h(\lambda_i)$ are equal the algorithm chooses the route and wavelength with higher $Cd(\lambda_i)$ value, that is the route and wavelength with more resources availability. Figure 29 shows the pseudo-code core of the *BAPHOR* algorithm for 2 hierarchical levels; Hn_A and Hn_B are respectively the length of the route A and route B in number of hops. As in the rest of proposed prediction-based algorithm of this Thesis the two-bit counters of the *PTs* are updated in order to learn. If the connection can be established the corresponding two-bit counter is decreased, otherwise it is increased.

8.5. Illustrative Example

Considering that every *RA* includes control functions with signalling capabilities, update messages are sent according to $N = 6$, i.e. every 6 changes. Then, a wavelength is defined as *POW* according to a percentage $p_r = 50\%$ (i.e., when the minimum number of available wavelengths on this link is lower than or equal to 3). For this illustrative example, it is assumed $B = 10$ fibres per link and 4 wavelengths per fibre. Suppose that incoming call requests arrive between nodes S and D in Figure 27.

Table 5. Precomputed shortest routes.

Source-destination pair	*Route A*	*Route B*
RA1-RA2	*RA1-RA2*	*RA1-RA3-RA4-RA2*
RA1-RA3	*RA1-RA3*	*RA1-RA2-RA4-RA3*
RA1-RA4	*RA1-RA2-RA4*	*RA1-RA3-RA4*
RA1-RA5	*RA1-RA3-RA5*	*RA1-RA2-RA4-RA5*

When a call request from node S (N1.1) to node D (RA5), in Figure 27, reaches node N1.1, this node applies the *BAPHOR* algorithm to select the lightpath based on the information represented in Table 6. Table 5 shows the 2 precomputed shortest and link disjoint routes, A and B, in N1.1 between RA1 and the rest of routing areas. There are two routes from each node belonging to RA1 to the other routing areas and a two-bit counter for every route and wavelength from RA1 to the other routing areas. Table 6 shows the database of the node N1.1. This database has the complete topology information about RA1 (the number of available wavelengths of each colour in every link), as well as a two-bit counter for every route in the second hierarchical level to the rest of the network. Note that H_n is the distance in number of hops in the second hierarchical level.

Table 6. Database Table and Prediction Tables.

Link	λ_1	λ_2	λ_3	λ_4	*Route*	λ_1	λ_2	λ_3	λ_4	H_n
N1.1-N1.2 (Availability)	6	3	3	6	Two-bit Counters *RA1-RA2* (Route A)	2	1	0	3	1
N1.2-N1.3 (Availability)	2	3	6	0	Two-bit Counters *RA1-RA2* (Route B)	2	3	1	2	3
N1.3-N1.4 (Availability)	6	3	0	2	Two-bit Counters *RA1-RA3* (Route A)	2	3	0	2	1
N1.2-N1.5 (Availability)	6	2	0	1	Two-bit Counters *RA1-RA3* (Route B)	1	2	0	1	3
N1.5-N1.3 (Availability)	6	6	6	6	Two-bit Counters *RA1-RA4* (Route A)	0	0	1	3	2
N1.5-N1.6 (Availability)	0	7	3	3	Two-bit Counters *RA1-RA4* (Route B)	0	1	3	2	2
N1.6-N1.4 (Availability)	1	1	1	1	Two-bit Counters *RA1-RA5* (Route A)	3	1	2	0	2
N1.1-N1.7 (Availability)	6	3	1	6	Two-bit Counters *RA1-RA5* (Route B)	0	1	3	2	3
N1.7-N1.8 (Availability)	0	3	6	6						
N1.8-N1.4 (Availability)	6	6	0	6						
N1.4-*RA3* (Availability)	6	7	7	5						
N1.8-*RA2* (Availability)	5	6	7	5						

Table 7 illustrates the *Od* and *Cd* values of the first hierarchical level (into RA1) and the values of the two-bit counters of the two shortest routes selected between the source (node N1.1) and the destination (one node of the RA5). The degree of obstruction, *Od* , is computed taking account that the network state information is updated every 6 changes, and also assuming *pr*=50%. Then, a wavelength is defined as *POW* in a link when the number of available wavelengths is lower or equal to 3. In this case the hierarchical weight, $W_h(\lambda_i)$, of each wavelength is calculated adding the value of the $W(\lambda_i)$ of the first hierarchical level with the value of the two-bit counter of the lightpath through the second hierarchical level. Note that in Table 7 only wavelengths with availability are considered. The *BAPHOR* algorithm selects the wavelength and route that minimizes $W_h(\lambda_i)$, that is route B and λ_4.

Table 7. Computing the $W_h(\lambda_i)$ values for the *BAPHOR* algorithm.

Route A	λ_1 λ_2 λ_3 λ_4	H_n	λ_i *(Od,Cd)*	*W* (λ_1)	*W* (λ_2)
N1.1-N1.2-N1.3-N1.4-RA3 *(1st hierarchical level)*	2 3 0 0 (Availability)	4	λ_1(1,2), λ_2(3,3)	*2*	*4*
		H_n		*Counter*	*Counter*
RA1-RA3-RA5 *(2nd hierarchical level)*	3 1 2 0 (Two-bit counters)	2		*3*	*1*
				W_h =5	W_h=5

Route B	λ_1 λ_2 λ_3 λ_4	H_n	λ_i *(Od,Cd)*	*W* (λ_2)	*W* (λ_3)	*W* (λ_4)
N1.1-N1.7-N1.8-RA2 *(1st hierarchical level)*	0 3 1 5 (Availability)	3	λ_2(2,3),λ_3(1,1), λ_4(0,5)	*2*	*3*	*0*
		H_n		*Counter*	*Counter*	*Counter*
RA1-RA2-RA4-RA5 *(2nd hierarchical level)*	0 1 3 2 (Two-bit counters)	3		*1*	*3*	*2*
				W_h = 3	W_h = 6	W_h = 2

8.6. Performance Evaluation.

Once the proposed hierarchical network structure has been analyzed by the illustrative example presented above, the proposed algorithms are evaluated by simulation. Simulations are carried out on the topology shown in Figure 27, but unlike the illustrative examples, the configuration is a 5-fibre topology, with 16 wavelengths on all the fibres on all the bi-directional links. It is also assumed that nodes N1.1 and N1.7 of RA1 act as source nodes, while there are 2 destination nodes in RA4 and RA5 respectively.

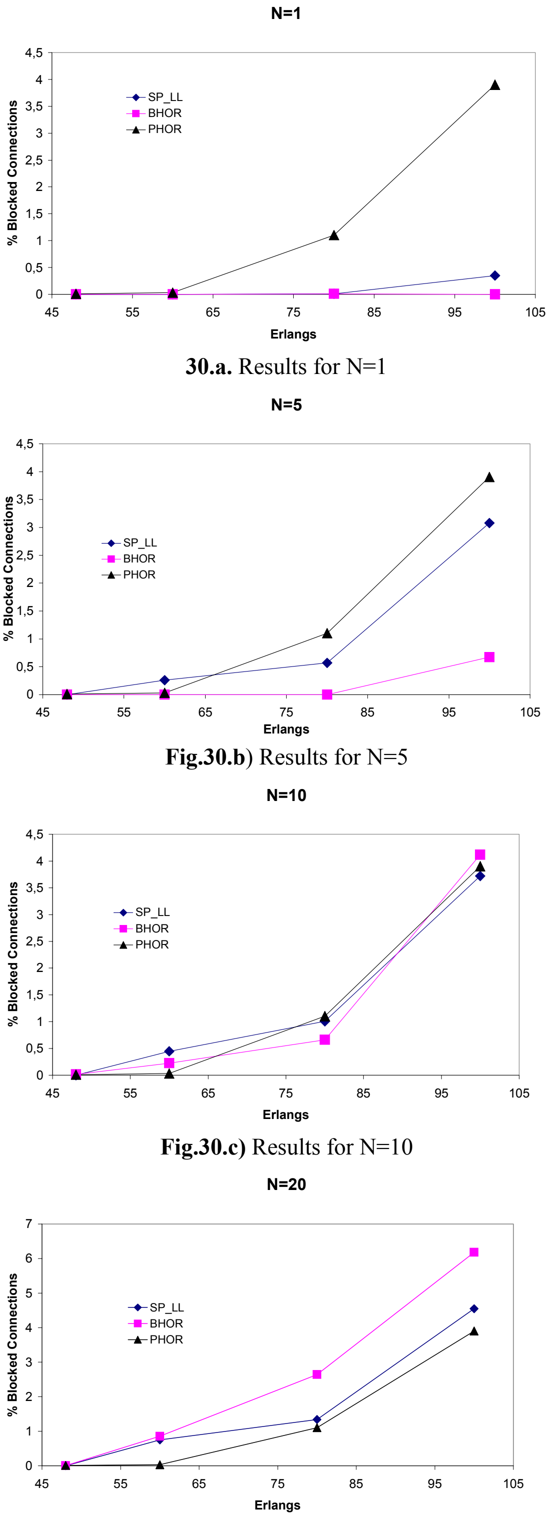
Connection arrivals are modelled by a Poisson distribution and the connection holding time is assumed to be exponentially distributed. The algorithms behaviour is measured in terms of percentage of blocked connections.

A. Preliminary Evaluation.

In Figure 30 it is evaluated the performance of the *PHOR*, the *SP-LL* (Shortest Path combined with Least Loaded with the aggregation scheme *NAS*), the *BHOR* (with the aggregation scheme NAS) and the *PHOR* in terms of the connection blocking probability. A set of simulations are carried out varying the traffic load between 48 and 100 Erlangs where the total number of connection requests is 20000 on each simulation run. All routing algorithms compute two shortest routes (A and B).

Results shown in Figure 30 are obtained for all the algorithms ranging the threshold updating *N*, between *N* = 1 (Figure 30.a), *N* = 5 (Figure 30.b.), *N* = 10 (Figure 30.c.) and *N* = 20 (Figure 30.d.). This *N* is not a periodical update, it *N* means the number of changes needed in the network to trigger an update. It is important to notice both the *PHOR* algorithm does not need update messages nor any aggregation scheme and the larger the *N* value the lower the signalling overhead. According to the results shown in Figure 30 the conclusions are the following. On the one hand, from the point of view of performance, for low values of *N* (i.e., *N* <=5) the *BHOR* is the algorithm presenting the lower blocking probability and the *PHOR* is the worst. This trend changes as the value of *N* increases. In fact, for *N* = 20, the *BHOR* exhibits the worst behaviour, while the *PHOR* is the best.

On the other hand, from the point of view of the signalling overhead and computation complexity, the *PHOR* is the best option since neither update messages nor aggregation schemes are required.



30.a. Results for N=1

Fig.30.b) Results for N=5

Fig.30.c) Results for N=10

Fig.30.d) Results for N=20

Figure 30. Connection blocking for the *SP-LL*, the *BHOR* and the *PHOR* algorithms.

Comparing the results obtained for the *BHOR* and the *SP-LL* algorithms in Figure 30, we conclude that while for $N = 1$ and $N = 5$ the *BHOR* behaves better than the *SP-LL* for all the traffic loads, the *SP-LL* exhibits better results than the *BHOR* for $N = 10$ and $N = 20$ with high traffic loads.

This is due to the special characteristics of the *BHOR* that uses the N when computing routes. As it is explained in subsection 7.4 the links of the routes having less than 50% of N available fibres determine the degree of obstruction of the route. In our simulations the number of fibres is 5 so that being for example $N = 5$ (the 50%, p_r, of N is 2), the links with 2 or lower available fibres contribute to the degree of obstruction. However, when $N = 10$ or $N = 20$, all the links are contributing to the degree of obstruction, since the number of available fibres is always lower or equal than 5 or 10 (computed according to p_r). In this scenario, the W factor becomes quadratic dependent with the number of hops in the route, and for this reason the algorithm tries to assign the shortest route. Because of such an assignment, when the traffic load is high this shortest route is heavily congested.

Figure 31 shows the results in percentage of blocked connections as a function of N, for different traffic loads. While the *SP-LL* and the *BHOR* algorithms behave worse than the *PHOR* algorithm (not affected by the N value) for high values of N, the best algorithm is the *BHOR* for low values of N.

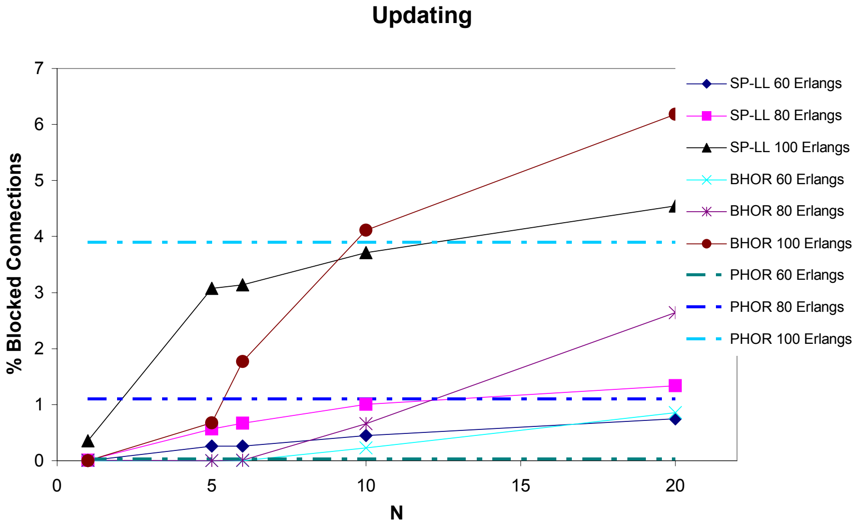


Figure 31. Connection blocking for the *SP-LL*, the *BHOR* and the *PHOR* algorithms depending on the updating frequency.

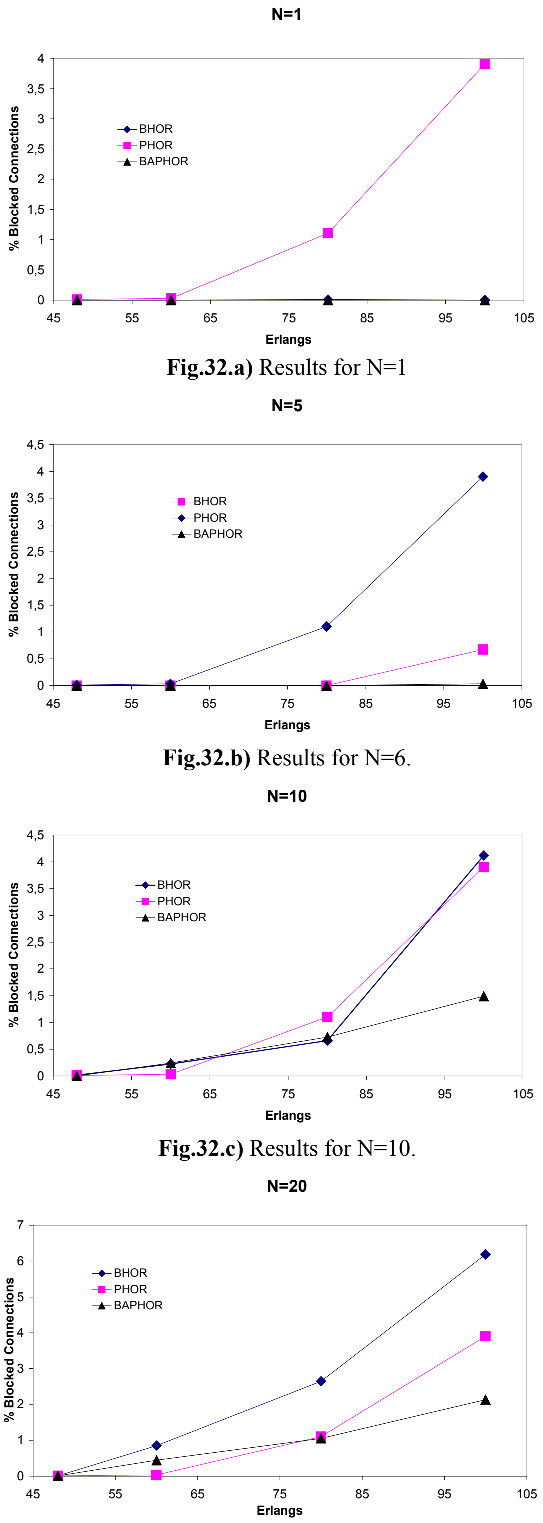


Fig.32.a) Results for N=1

Fig.32.b) Results for N=6.

Fig.32.c) Results for N=10.

Fig.32.d) Results for N=20.

Figure 32. Connection blocking for the *BHOR*, the *PHOR* and the *BAPHOR* algorithms.

B. Global Evaluation.

In order to evaluate the *BAPHOR* algorithm performance the blocking probability produced by the *BHOR* (*NAS*), *PHOR* and *BAPHOR* algorithms are compared when traffic load is ranged from 0 to 100 Erlangs. The results for the *BHOR* and the *BAPHOR* algorithms are presented for 4 different values of *N*, *N* = 1 (Figure 32.a), *N* = 6 (Figure 32.b), *N* = 10 (Figure 32.c) and *N* = 20 (Figure 32.d). Note that the *PHOR* does not vary with the *N* value since it does not need any update messages. Notice that the *BAPHOR* algorithm better tolerates high values of *N* than the *BHOR*. This is because the routing decision is carried out also including prediction issues. Moreover, for 48 Erlangs all the algorithms has similar performance. Instead, from 50 Erlangs the connection blocking strongly depends on the value of *N*. However, after analyzing all the graphs included in Figure 32 the lower connection blocking is obtained by the *BAPHOR* algorithm. This is justified because the *BAPHOR* algorithm combines the benefits of both the *PHOR* algorithm, i.e., prediction issues, and the *BHOR* algorithm, i.e., load balance and congestion reduction.

Finally, Figure 33 shows the connection blocking behaviour for the *BHOR*, *PHOR* and *BAPHOR* algorithms as a function of the value of *N*. While the *BAPHOR* algorithm behaves similarly than the *BHOR* algorithm for low values of N, the *BAPHOR* (and also the *PHOR* algorithm) better tolerates high values of N for high traffic loads compared to the *BHOR* algorithm.

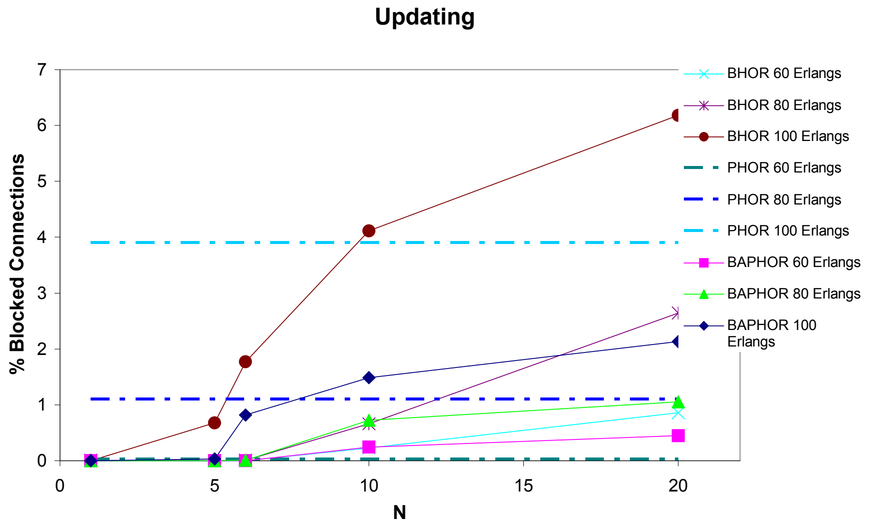


Figure 33. Connection blocking for the *BHOR,* the *PHOR* and the *BAPHOR* algorithms depending on the updating frequency.

9. The Prediction-Based Routing Mechanism in Multi-layer Networks.

9.1. Motivation.

The new advances in Optical-Cross Connects (*OXC*s) will bring an increase in switching flexibility in Optical Transport Networks (*OTNs*). Properly configuring the *OXC*s allows individual wavelength channels to link consecutive fibres into an end-to-end lightpath. As manual intervention by network operators is currently needed for provisioning a lightpath, this can take up to a couple of weeks or even months. The high dynamism of traffic patterns however will require that *OTNs* react within a sufficiently short time frame. As such, research is currently focusing on the development of Automatically Switched Optical Networks (*ASONs*). For example, in an *IP*-over-*ASON* scenario, the lightpaths are used to create links in the *IP* network topology, which is however completely independent of the physical optical topology. An automatic circuit-switched optical network allows lightpaths to set up and tear down dynamically bypassing the manual intervention, using User Network Interface signalling, as standardized by the *OIF* [70]. For an *IP*-over-*ASON* network scenario, these lightpaths provide the bandwidth that connects the *IP* routers together.

Implementing this fast-responding *ASON* functionality will allow direct links to be created or removed in the logical *IP* topology, when either extra capacity is needed, or existing capacity is no longer required. Reconfiguring the logical topology constitutes a new manner by which Traffic Engineering (*TE*) can solve or avoid network congestion problems and service degradations. As both *IP* and optical network layers are involved, this is called the Multilayer Traffic Engineering (*MTE*), proposed in [71] by Puype et al from the IBBT (Interdisciplinary institute for BroadBand Technology).

In the *MTE* strategy, the logical *IP* network topology is reconfigured dynamically according to the traffic pattern at hand. One bandwidth request in a node in the *IP* layer, and the corresponding establishment implies one or more setups of lightpaths in the optical layer. This characteristic of the *MTE* traffic makes the *MTE* performance closely dependent on the updating frequency of the network state information, in terms of connection blocking. In this scenario the updating frequency needed to keep the network state information updated is physically unaffordable. Hence, with usual *RWA* algorithms most of the routing decisions are performed using inaccurate network state information. The solution proposed in this Thesis is to apply the *PBR* mechanism in the optical layer of the *MTE*. When using the *PBR* in the optical layer of the *MTE* strategy the source nodes do not receive any update messages about which links have been set up or torn down, but they can learn the network state from previous connections requests.

9.2. Review of the *MTE* strategy

The purpose of Multi-layer Traffic Engineering (*MTE*) [71] is to extend “classic” traffic engineering with cross-layer capabilities, using the newly found flexibility available in next generation Automatically Switched Optical Networks (*ASON*) [72]. *MTE* does this by reconfiguring the logical topology in the *IP* layer, setting up and tearing down optical connections which support *IP* links. Apart from this logical topology configuration, the *MTE* strategy also has to route the offered traffic over the logical topology and of course, both routing and topology configuration are influenced by each other.

The *MTE* strategy will be used to route offered traffic into the *IP* layer and also reconfigure its logical topology. This results in connection requests towards the optical layer where they are to be routed by a *RWA* algorithm. The *IP* traffic between two *IP* routers is conceived as a flow which has variations on both a large and short time scale. The large time scale variations are achieved by periodically adjusting its average bit rate as a random uniformly distributed variable. The additional short time scale fluctuations resemble smaller changes in bit rate as seen in a Poisson arrival process; they were generated using a Markov chain for tractability.

These traffic flows (one for each *IP* router pair) will serve as input for the *MTE* strategy, which is based on the concept described in [73] by Puype et al, where it is presented a strategy which can reduce a full-mesh in the *IP* layer towards a sparser dynamic logical topology through appropriate *IP/MPLS* routing and fast optical layer connection set up/tear down. However, since a *IP* traffic flow in this discussion now will have a bit rate higher than a optical connection’s bandwidth, the strategy in [73] has been extended to cope with these higher bandwidths, and to allow multiple parallel optical connections (*IP* links) between two *IP* routers (or optical end-point nodes).

The basic concept of the *MTE* strategy is to start from a virtual full mesh in the *IP* layer as logical layer, and to use multi-hop routing in trying to reduce the number of *IP* links actually carrying traffic. For this goal, the *MTE* strategy uses an *IP* layer cost function depending on load of the *IP* links. It is formed such that routing over *IP* links with a low load will be avoided, eventually diverting all traffic away from such links, allowing it to be dropped from the starting full mesh. The cost function is used in routing flows over a virtual full mesh (serving to express the high flexibility in connection setup of the underlying optical layer), and once a flow's new route is determined, it is then rerouted using *IP/MPLS* in the actual logical topology. Also, when as a result of those reroutes the actual logical develops *IP* links that no longer carry traffic, they will be torn down. Likewise, if the new routes require *IP* links not set up, they will be requested to the optical layer.

A. Cost Function

The *MTE* strategy and its cost function allows multiple optical connections between a single router pair, in order to allow larger amounts of *IP* traffic, more interesting grooming constraints and higher optical layer connection load.

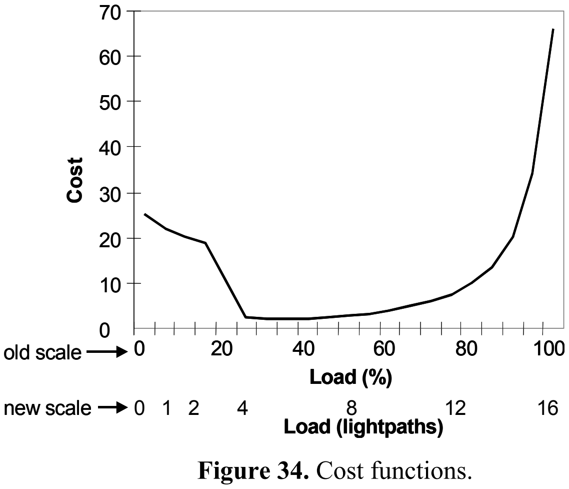


Figure 34. Cost functions.

Optical connections could be concatenated into a single *IP* bandwidth pipe. This also means that all *IP* traffic between two routers still follows the same *IP* route – although optical route may differ because of splitting in separate optical connections, depending on the *RWA* algorithm used.

On Figure 34 some sample *MTE* cost functions for a maximum load of 16 optical connections (1600% of requested bandwidth). Accordingly, *IP* links can consist of 1 up to 16 optical connections.

The function then is characterized by three parameters. Firstly, there is a High Load Threshold – *IP* links with a load above HLT receive an exponentially rising cost. However, also lightly loaded links (with a load below Low Load Threshold, LLT) are penalized with a higher cost. The higher cost for low loads is defined against the cost for moderate loads by the Low/Moderate Ratio (LMR), indicating the ratio between cost for low loads (LC) and cost for moderate loads (MC); LMR = LC/MC. This cost penalty avoids establishing many and thus inefficiently used links, and thereby promotes grooming of traffic into *IP* links carrying a bundle of flows.

B. Capacity adjustment mechanism

Allowing *IP* links consisting of multiple parallel lightpaths brings with it three important requirements for the *MTE* strategy. First, as it has been presented above, there is a necessary adaptation of the cost function to these higher loads. Also, the concatenation requires a capacity adjustment scheme, and finally there is some choice in optical connection tear down for this scheme.

IP links can now have a load of several lightpaths. This allows the network to cope with larger traffic demands. Traffic however may still be erratic, so the actual load of an *IP* link in number of required optical connections may fluctuate. Therefore, a capacity adjustment mechanism was added to the *MTE* strategy, which uses fast optical connection setup to deliver bandwidth on demand to an *IP* link. This way, optical bandwidth can be used more efficiently (not having to set up a more static maximum amount of optical connections per *IP* links).

Also, the bandwidth adjustment scheme can be used to take care of the fast fluctuations, not having to rely on rerouting or logical topology reconfiguration in these cases. Therefore the *MTE* strategy now has three mechanisms operating at different time scales.

First there is the logical topology configuration, where *IP* router adjacencies are changed (e.g., hours between updates). Here, the cost function attracts or diverts traffic such that new *IP* links become necessary or some existing *IP* links can be removed. The time between topology updates coincide with the interval between the long term traffic flow bandwidth changes (uniform random distributed). Second, there are *IP/MPLS* reroutes, periodically changing *LSP* routes over the logical topology (e.g., possibly tens of minutes between reroutes). For this, the cost function is also used; in fact, for simplicity, the routes are fixed for each logical topology.

Lastly, there is the *IP* link capacity adjustment, where the optical connections between adjacent *IP* routers are added (or removed) on-the-fly (sub-second timescale). Their timescale corresponds with the short term (Poisson process governed) traffic fluctuations.

Of course, in an actual network, the traffic has to be actually measured and not generated, which can lead to several problems on its own, as described by Yan et al in [74], where it is examined the influence of the length of the observation window of traffic measurements on performance. Note that both the logical topology update and capacity adjustment are cross-layer traffic engineering techniques. The first will have a larger impact on *IP* layer performance, whereas the latter is mostly transparent, but relies on fast optical setup and teardown times.

C. Optical connection selection

The addition of the capacity adjustment scheme brings with it much more frequent optical connection setup and teardown. When a new optical connection is needed, it is simply requested from the optical layer (assuming sufficient available optical capacity).

However, since *IP* links now consist of a bundle of optical connections, there is some

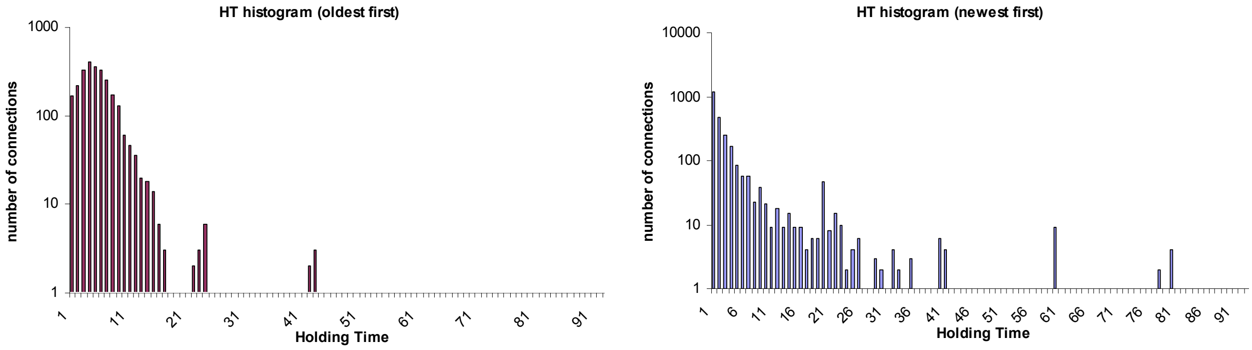


Figure 35. Impact on optical connection holding time distribution.

choice in connection tear down during *IP* link downgrade. Three options have been examined. The 'newest-first' strategy will tear down the newest (last set up) lightpaths first, keeping long-term lightpaths in the networks. The 'oldest-first' strategy does the opposite and will then spread out the distribution of call duration of lightpaths, avoiding optical connection with very short holding times, hopefully limiting optical connection dispersion. The 'random-first' strategy is situated somewhere in-between, obviously. Figure 35 shows the impact of optical connection tear down selection strategy on the holding time distribution.

The distribution for the 'oldest-first' strategy is much more compact, while the 'newest-first' distribution has a high mass at very short optical connections (as expected). In this case, the time axis is 1 unit per bandwidth adjustment period. Furthermore, logical topology updates were performed every 20 time units. One notices the spikes in the distribution every 20 units (especially in the 'newest-first' case, where long-term optical connections are promoted), corresponding with the logical topology update lightpath requests.

9.3. *PBR* in the *MTE* strategy.

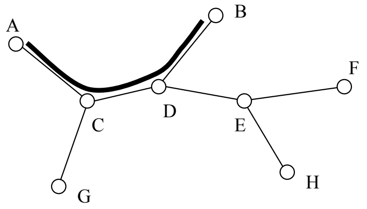
The original *MTE* strategy presented in [73] uses shortest path first routing in the optical layer. This means that an optical connection between two optical nodes / *IP* routers has a fixed path and there is no wavelength assignment. Moreover, the *MTE* strategy does not consider the possibility of blocking in the optical layer of the *MTE*. That is, up till now, the *MTE* strategy considered that the number of wavelengths in every path of the physical topology is unlimited. These assumptions did allow minimizing total capacity usage as a performance parameter. However, the number of blocked connections in the optical layer is not a parameter to be minimized because of the unlimited number of wavelengths. As it is presented above one of the parameters to adjust is the number of lightpaths per *IP* link. The number of optical connections per link should be medium with neither over nor low loaded links. For all the reasons exposed above, in the optical layer of the *MTE* there was not any *RWA* algorithm implemented.

When the number of wavelength is limited it is necessary to implement a *RWA* algorithm that properly assigns routes and wavelengths. Also, when limiting the number of wavelength the routing inaccuracy problem appears.

Since the set up and tear down of a lightpath on the optical network affects the free and used capacity on several optical links, all node pairs with a Shortest Path (*SP*) over these links have to have their load information recalculated (reflooded, etc.). See Figure 36 for an example. Here the set up of a relatively short lightpath A-B affects the majority of the node-pairs. This gives quite a lot of overhead. Using the current strategy as depicted in the figure above for more than one shortest path per node-pair would result in an exponential amount of maximum flow calculations each time an optical action is performed which is consequently not scalable.

When the *MTE* strategy is extended [74] and an *IP* link can consists of 1 up to 16 optical connections, it means that every new bandwidth requests in the *IP* layer can be up to 16 optical connection requests in the optical layer of the *MTE* strategy. With these traffic characteristic, the performance in terms of blocking probability becomes more dependent of the updating frequency when using typical *RWA* which needs flooding of the network state information. Now, one bandwidth request in a node in the *IP* layer, and the corresponding establishment imply a lot of reconfigurations (set up of a lot of lightpaths) in the optical layer, and thus a lot of signalling overhead. In this scenario it would be appropriate to use mechanisms independent of the flooding of network state information such as the *PBR*.

It might be interesting to reduce the flooding and calculation times in the optical layer by replacing the simple *SP* with a Prediction Based Routing (*PBR*) mechanism, additionally using for example k-shortest paths for each optical node pair, or even using all possible paths.



New lightpath A-B

Affected load metrics (20)
A-[anything]; B-[anything];
C-D; C-E; C-F; C-H; D-G; E-G; F-G; H-G

Unaffected load metrics (7)
C-G; D-E; D-F; D-H; E-F;
E-H; F-H;

Figure 36. Effect on node-pair load metrics for setting up a new lightpath.

As mentioned above, using the *PBR* in the optical layer is to do optical routing where the source nodes do not receive any (or minimal) update messages about which links have been set up or torn down, but they can learn optical layer network state by keeping track of whether previous requests have been blocked or not.

9.4. Performance Evaluation.

The performance (in terms of percentage of blocked connections in the optical layer) when applying the Routing and Wavelength Prediction (*RWP*) algorithm as the *RW*A algorithm is compared to the performance of an usual *RWA* algorithm, such as the Shortest Path combined with the First-Fit, *SP-FF*.

Simulations are carried out on the topology shown in Figure 21. The *RWP* algorithm selects the shortest lightpath among all the possible lightpaths having their Wavelength Route Counters (*WRC)* lower than 2 and with output link availability.

The *RWP* performance is evaluated by comparing its behaviour against that obtained by a usual *RWA* algorithm requiring updating of the network state information, such as the Shortest Path combined with the First Fit (*SP-FF*). In this case, the algorithm selects the shortest lightpath among all the possible routes with availability in all the links. Note that now, it is necessary to update the network state information to know link availability along the routes. If there is more than one shortest route it selects the more available, i.e., with more available wavelengths. This algorithm is simulated ranging the time between updating, that is, the time that the network state information is flooded through the network, between 0 and 20 units of time. Updating every 0 units of time represents the ideal case (complete accuracy); at any point in time the source nodes know the entire network state. This is an ideal case because it is not only unaffordable from the point of view of signalling overhead, but also it is physically impossible. Finally, nodes in Frankfurt, Madrid and Oslo are assumed acting as source and destination nodes. Figure 37 shows the results in percentage of blocked connections of the *RWP* algorithm and the *SP-FF* combined with the three policies used to tear down the optical connections, “newest first”, “oldest first” and “random first”.

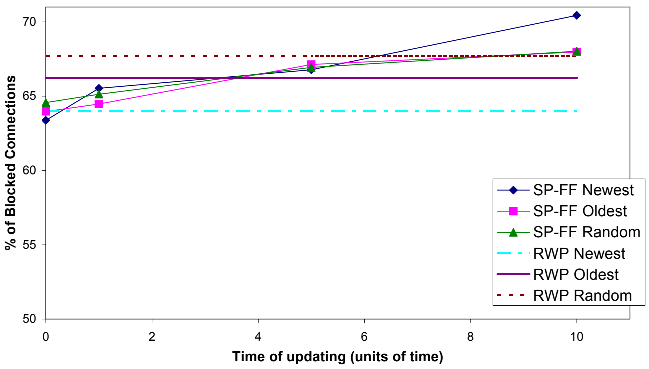


Fig.37.a) Results for 4 wavelengths.

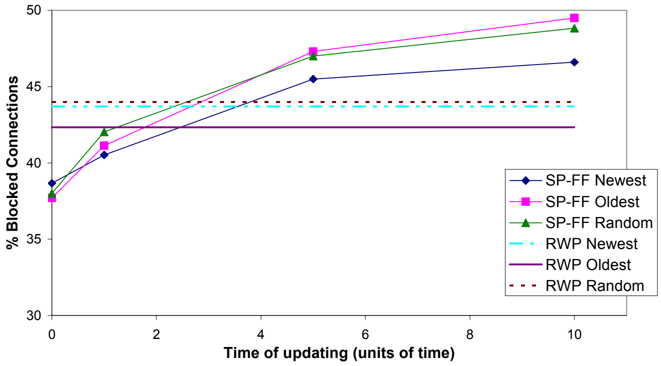


Fig.37.b) Results for 8 wavelengths.

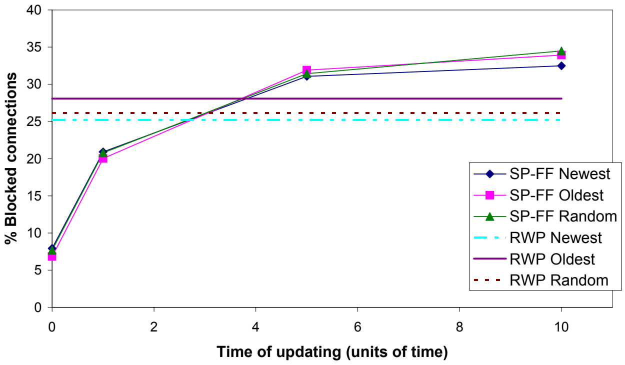


Fig.37.c) Results for 16 wavelengths.

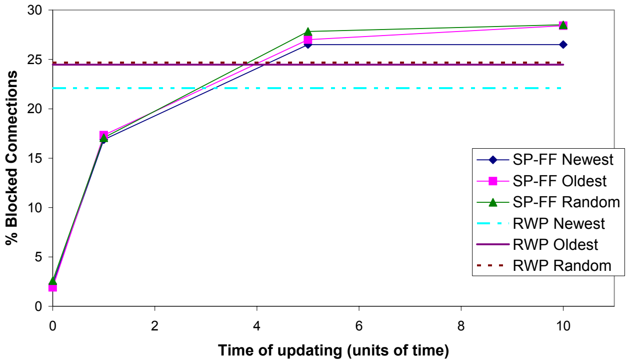


Fig.37.d) Results for 20 wavelengths.

Figure 37. Percentage of Blocked Connections versus Time of updating for *RWP* and *SP-FF*.

Results for four different network resource configurations are showed. All the links of the network are one-fibre links and the number of wavelengths in each link can be 4, 8, 16 and 20 wavelengths, for the different configurations.

When the resources of the network are scarce or even insufficient for the *MTE* traffic requirements, i.e., 4 wavelengths per link, (Figure 37.a), the minimum percentage of blocked connections is achieved by the *SP-FF* (combined with "newest policy") algorithm with ideal updating. The *SP-FF* algorithms ("newest", "oldest" and "random") rapidly increase their percentage of blocked connections when the time between updating increases. Even when the updates occur each unit of time, the percentage of blocked connections increases one 2%, 0,5 % and 1% for the "newest", "oldest" and "random" policies respectively. Due to the *MTE* traffic characteristics, every unit of time various optical connections can be set up or torn down by the source nodes. Updating every unit of time implies that some source nodes select routes with out-of-date network state information, because at the same time other source nodes can be setting up or tearing down lightpaths. The *RWP* algorithm does not vary with the updating frequency because it does not need updating of the network state information. In Figure 37.a) the *RWP* with the "newest" policy has better performance than the *SP-FF* (newest) with updating every 1 units of time. Also, comparing the three tear down policies, for the *RWP* algorithm the best results correspond to the "newest first" policy and the worst for the "oldest first" policy.

Results for 8 wavelengths are shown in Figure 37.b). For this resource configuration the *RWP* algorithm outperforms the *SP-FF* algorithm even with updating every 5 units of time. The best policy for the *RWP* is the "newest first" too.

Results for 16 wavelengths are shown in Figure 37.c). The results of the *SP-FF* algorithm combined with the three torn down policies are very similar. For ideal updating the percentage of blocked connections produced by the *SP-FF* algorithm with the three torn down policies is approximately 7%. But if the updating time is increased the source nodes use inaccurate information and the percentage of blocked connections rises. Between 0 and 1 updating time units the percentage of blocked connections rises one 13%. And between 1 and 5 updating time units the percentage of blocked connection rises one 11% more, resulting approximately one 31%. On the other hand the *RWP* only has between 25-28% of blocked connections and the best policy is the "newest" and the worst the "oldest". Finally, for 20 wavelengths (Figure 37.d) the blocked connections for the *SP-FF* ideal case are approximately one 2%, being 17% and 27% when the updates occur every 1 or 5 units of time respectively. The *RWP* has results between 22 and 24% of blocked connections. Note, that for this network configuration the best results for both algorithms correspond to the “newest” policy and the worst for the “random” policy.

Summarizing, the *RWP* results outperform the *SP-FF* results with updating every 5 units of time, when there is 8, 16 or 20 wavelengths. Even when the resources are limited (4 wavelengths), the *RWP* algorithm has a lower percentage of blocked connections than the *SP-FF* updating every 1 units of time.

Due to the *MTE* traffic characteristic the results for usual *RWA* algorithms that need network state information are very dependent on the updating frequency. For this reason the *MTE* strategy works better with *RWA* algorithms that do not base their decision on inaccurate information, but on predicted information. In Figure 38 there is represented the accumulative traffic load between the source-destination pair Madrid-Frankfurt sorted by connection request number for the three policies and for the first 500 connection requests. The policies “oldest first” and “random first” produce a more regular traffic load pattern, but the “newest first” policy produces a more irregular traffic pattern. Also, the “newest” has high total traffic load at the beginning which diminishes during later connection requests, because around request 500 it has more or less the same mean traffic load as the other two policies. This characteristic of the “newest” policy makes it very suitable for the *PBR* mechanism. At the beginning, with high traffic load, the data structures of the *PBR*,

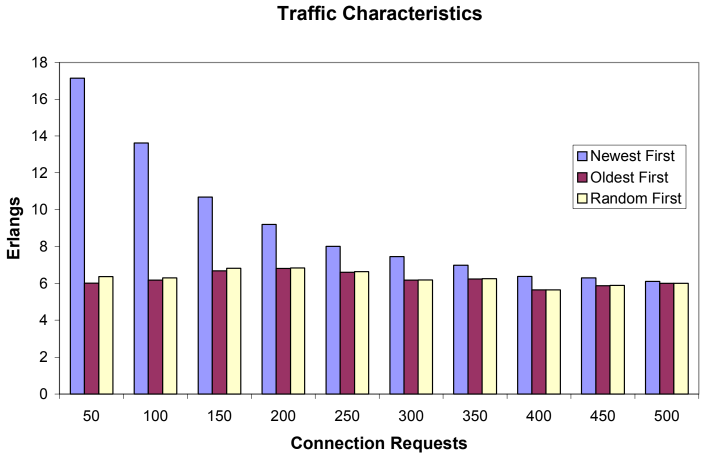


Figure 38. Accumulative traffic load for the different tear down policies.

Wavelength Route Counters, can be trained, learning from the blocked connections produced. Then, when the traffic load is medium, the route counters have learned the best route for every connection request. That is, the *PBR* mechanism requires that there are blocked connections in order to learn. This explains the better results of the “newest” policy of the *RWP* algorithm for 4, 16 and 20 wavelengths. Only for 8 wavelengths the best results are for the “oldest” first torn down policy.

PART III

IP/MPLS Networks

In this part it is summarized some of the recent work of routing in *IP/MPLS* networks. Before describing the *PBR* mechanism for *IP/MPLS* networks, they are reviewed other predictive schemes applied to network scenarios. The *PBR* mechanism is deeply described as a predictive routing scheme; different algorithms are proposed, illustrated and evaluated by simulation.

10. *QoS* Routing in *IP/MPLS*

QoS routing consists on selecting the most appropriate path that fulfils the *QoS* requirements, for example bandwidth or end-to-end delay. There are a large number of contributions in this topic that a reader can found in the literature. Just to define the scenario we present a short review of some of them.

Guerin and Orda proposed the Widest Shortest Path (*WSP*) algorithm in [75]. This algorithm selects the widest path, that is, with more available bandwidth, among the shortest paths in hop count. Authors present three versions of the algorithm, the first one selects the path among a set of exact precomputed paths. The second algorithm computes paths on demand using a modified Dijkstra's algorithm. And the third algorithm selects the path among a set of approximate precomputed paths.

The Shortest Widest Path (*SWP*) was proposed by Wang and Crowcroft in [76] and it selects the shortest path in number of hops among the widest paths. Authors associate the length of the path in number of hops with the end-to-end delay. For this reason the algorithm selects the minimum delay path among the paths with more available bandwidth. The *SWP* was proposed as a centralized source routing algorithm and also for distributed hop-by-hop routing algorithm.

The Minimum Interference Routing Algorithm (*MIRA*) was proposed by Kodialam and Laksham [77]. The main idea underlaying the *MIRA* algorithm is to select paths that do not interfere "too much" with other paths that can satisfy future traffic demands. In principle this problem is NP-hard but authors developed a heuristic path selection. In short, the algorithm selects the path that maximizes the minimum maxflow between all other source-destination pair of nodes. The maxflow is defined as the upperbound on the total amount of bandwidth that can be routed between a source-destination pair of nodes. First of all, the critical links are defined as the links that whenever a path is set up over those links the maxflow of one or more source-destination node pair decreases. Then, a shortest cost path algorithm (Bellman-Ford or Dijkstra) is used to compute the path in a graph where the links are weighted depending on their "critically".

The algorithms presented in [78] by Suri et al are based on the *MIRA* algorithm. The Profile-Based Routing (*PBR*) algorithm needs any knowledge about the traffic distribution on the source-destination node pairs of the network. The main difference with *MIRA* is that the *PBR* uses a “traffic profile” of the network obtained by measurements as a rough predictor of future traffic distribution. This profile is used in a pre-processing step to determine certain bandwidth allocations on the links of the networks. The offline of this pre-process is used to guide the online algorithm, and imposes traffic admission control by rejecting some requests because of their future blocking effects in the network.

In [79] Yang et al presented a work also based on the *MIRA* algorithm. The algorithm proposed computes the delay-weighted capacity (*DWC*) for each source-destination pair of nodes. To compute the *DWC* first of all, the paths between a source and destination node pair are computed and ordered according to the path delay as follows. The least delay path is computed and then the links of this path are pruned of the network graph. In this new network graph the least delay path is computed again, and this will be the second least delay path of the set. The process is repeated until no paths can be found in the remaining network graph. Once the least delay set of paths is defined, the *DWC* can be computed. The *DWC* of a source-destination pair of nodes is defined as the weighted sum of the bandwidth of the paths of the previous set of paths. The weights are inversely proportional to the end-to-end delay value of the paths. Then, the critical links are defined as the links whose inclusion in a path produces that the *DWC* of several other paths decreases. As in the *MIRA* algorithm the links are weighted according to their “critically”; and finally a shortest cost path algorithm selects the path with least cost.

A different approach to the *QoS* routing with bandwidth constraint can be found in [80]. Khan an Alnuweiri proposed the Fuzzy Routing Algorithm (*FRA*) that is a modification of the shortest cost path Dijkstra’s algorithm that uses fuzzy-logic member-ship functions. Fuzzy optimization allows mapping values of different criteria into linguistic values that characterize the level of satisfaction with the numerical value of the objective. First of all, the links of the graph without enough bandwidth to satisfy the requested bandwidth are pruned in the network graph. The next step of the algorithm computes the path feasibility according to a fuzzy criterion. The criterion used is the node reachability and is defined by means of a linguistic rule. A fuzzy logic linguistic rule is an IF-THEN rule. The rule used to evaluate the reachability is the following: IF a path to node y through node x has low bandwidth utilization on bottleneck link AND path to y trough x has low bandwidth utilization on links other than the bottleneck link AND path to y through x has less number of hops THEN the node y is reachable. Then, the path selected is the path through which the destination has most reachability. Authors use Ordering Weighted operators to represent the AND and OR functions. These operators allows the adjustment of the degree of AND and OR.

A more detailed review of *QoS* routing can be found in [81].

10.1. The Routing Inaccuracy Problem in *IP/MPLS* Networks

In this section some of the *QoS* routing solutions that take into account the routing inaccuracy problem are reviewed. In [82] Guerin and Orda study the path selection using inaccurate or imprecise network state information subject to bandwidth constraint or end-to-end delay separately. Authors conclude that with bandwidth constraint the problem is polynomial solvable and the paths can be computed using relatively standard algorithms. However with end-to-end delay constraint the solution is NP hard and only approximate solutions can be found. Authors present two different approaches to find these solutions, the rate-based approach and the delay-based approach. Authors show that for the rate-based approach the problem is intractable. However they find that there are some special cases that have tractable solutions. For the delay-based model, instead the problem is also intractable; it can be solved using some heuristic based on dividing the end-to-end delay constraint into local delay constraints. These heuristic solutions can be applied with reasonable complexity when the source of inaccuracy is the aggregation process involved in a hierarchical network.

In [83] Lorenz and Orda investigate the impact of inaccurate information with end-to-end delay requirements. They propose the routing problem and propose the optimally partitioned most probable path *OP-MP* solution. Authors assume that the delay values that are advertised by each link are random variables with known distributions. The end-to-end delay constraints are decomposed into local delay constraints, thus, one part of the solution is to compute the optimal partition. To solve the *OP-MP* problem is to find a path accomplishing the set of partitioned delay constraints. Authors find pseudo-polynomial solutions for a wide class of probability distributions of the delay that every link advertises.

Apostolopoulos et al propose the safety-based routing [84] which incorporates knowledge of the underlying inaccuracy on computing a safety path under bandwidth requirements. Safety of a link is defined as the probability that the requested bandwidth is available on the link. Authors divide the inaccuracy in two types, quantifiable inaccuracy and inaccuracy arbitrary. With a triggering policy it is possible to infer a reasonable range for the actual link metric at any instant, this is the quantifiable inaccuracy. However with a large hold down timer the inaccuracy is arbitrary and there is no an explicit relation between the actual value and the last advertisement. The safety-based routing computes the range of feasible values for the actual available bandwidth on a link given the requested amount of bandwidth, the last advertised value and assuming a triggering policy, that is quantifiable inaccuracy. This is done assuming that the bandwidth values are uniformly distributed. The safety of a path is then the product of the safeties of its links. The two proposed algorithms that use the safety information to select the path are the safest-shortest route and the shortest-safest route. The former selects the path with the largest safety value among the shortest paths. And the later selects the shortest path among the paths with the largest safety. Also the safety is included when computing the paths in the topology graph. In usual routing algorithm like the widest-shortest path the links that do not have enough bandwidth according to the last advertisement are pruned when computing the shortest path. However in safety routing the links which have to be pruned when computing the safest or the shortest route in the graph depend on a cut-off value. The cut-off value corresponds to the degree of safety of the link and the range corresponds to s=0 to s<1. According to this, the links pruned on the graph depends on the selected cut-off value. By simulation authors show that shortest-safest is the most effective safety-routing algorithm for all type of triggering policies, and assuming uniform distributions of advertised values of real bandwidth. Safety-routing is also beneficial when moderate hold-down timers are used (inaccuracy arbitrary).

The ticket-based routing is proposed in [85] by Chen et al, which uses multipath distributed routing algorithms to find a low cost feasible solution. The source nodes send probes (routing messages) carrying one or more tickets to find a low cost path that satisfied the delay or bandwidth requirements. There are two classes of tickets, the yellow and the green tickets. The yellow tickets prefer paths with smaller delays (in the case of delay constraint) or with more available bandwidth (in the case of bandwidth constraint); and the green tickets prefer paths with low cost. The algorithm utilizes the inaccurate or imprecise state information in the intermediate nodes to guide the tickets along the possible best paths. The total number of tickets sent determines the signalling overhead produced by this mechanism. Authors solve both problems finding the optimal number of tickets and distributing these tickets between yellow and green tickets, by utilizing the inaccurate network state information. When a ticket finds a link that does not accomplish the *QoS* requirement (bandwidth or delay) is invalidated. The process finishes when all the tickets are received by the destination node. If only invalidated tickets are received it is because there is no feasible path, and if only one valid ticket arrives there is only one feasible path. However when more than one valid ticket is received the path with the least cost is selected. This is possible because the probes carrying the tickets accumulate the cost of the path they traverses.

Masip et al present in [87] an enhancement of the Bypass-Based Routing (*BBR*) mechanism introduced in [86]. The main characteristic of the *BBR* mechanism is that if the algorithm selects a path that really cannot cope with the bandwidth requirement, this path is not rejected. Instead, the *BBR* mechanism tries to skip those links that do not have enough bandwidth by using precomputed bypass paths. The basis of the *BBR* mechanism is a new parameter introduced in the path selection to represent the routing inaccuracy, the Obstuct-Sentitive-Links (*OSLs*). A link is *OSL* when potentially will not have enough resource to support the traffic requirements. A link is defined as OSL if the requested bandwidth, b_{req}, belongs to the range generated by the last advertised value of bandwidth in this link. Note that this range is different depending on the triggering policy.

The two proposed algorithms in [86] are the *SOSP* (Shortest-Obstruct-Sensitive Path) and the *OSSP* (Obstruct-Sensitive-Shortest-Path). The former selects the shortest path among the paths with the minimum number of *OSL* links. The later selects the path with the minimum number of *OSL* links among the shortest paths. If more than one path there exists both algorithms select randomly one. Once the route is selected the BBR mechanism computes an alternative path that bypasses those OSL links. The bypass-path it is selected by using links that bypass the *OSL* links and that cannot be OSL.

In [87] an enhancement of the *BBR* mechanism that tries to balance the path length and the residual bandwidth when selecting the path is presented. The two proposed algorithms are the *WSOSP* (Widest-Obstruct-Sensitive-Path) and the *BOSP* (Balanced-Obstruct-Sensitive-Path). The *WSOSP* is a modification of the *SOSP*. In both the path selected is the shortest path among the path with the minimum number of *OSL* links. However in *WSOSP*, when the final selection includes more than one path, the path is not randomly selected, instead the widest, with more available bandwidth, path is selected.

Therefore the *BOSP* tries to balance the path selection process, avoiding those paths that are both widest but to long and shortest but to narrow. As in *SOSP* and *WSOSP* the shortest path among the paths with less *OSL* links is selected. But when in the final selection there is more than one path, the path selected is that minimizing the F_p parameter. F_p is calculated according to: $F_p = n \cdot [max(1/b^i)]$, being n the number of hops of the path, being b^i the available residual bandwidth in link i of the path, and ranging i from 0 to the number of hops of the path.

Unlike the previous works in [88] Korkmaz et al study the problem of finding a path subject to both bandwidth and delay constraints under inaccurate network state information, that is, a multiobjective problem. The problem is solved by means of a probabilistic approach, and as in [82]and [83] it is reduced to find the most probable path that satisfies the bandwidth and delay constraints (*MP-BDCP*). First of all the problem is divided into the *MP-BCP* (most probable bandwidth constrained path) and the MP-DCP (most probable delay constrained path). The first, *MP-BCP*, has an exact polynomial-time solution, and authors propose a modified version of the Dijkstra's algorithm to solve it. The second problem, *MP-DCP* is NP-hard and authors propose approximate solutions for two cases of the problem. The first solution corresponds to the case that there exists one path that has a mean delay lower or equal than the constrained delay. In this case, the algorithm selects a path that minimizes both the mean and the variance delay, running in average 3 times a modified Dijkstra's algorithm. For the second case, there is no path with mean lower or equal than the constrained delay, the algorithm select the path that minimize the mean delay while maximizing the variance delay, running in average two times a modified Dijkstra's algorithm. The complete problem, *MP-BDCP*, is to find a path that maximizes both, the probability that the path accomplishes the bandwidth constraint and the probability that the path accomplishes the delay constraint. The proposed solution, first computes two paths maximizing the probability that the path accomplishes the bandwidth requirement and that accomplishes the delay requirement respectively. If the two paths are the same, this is the optimal solution. If the paths are not the same the *MC-DCP* solution is called iteratively computing a set of nondominated paths. The probability that the path accomplishes the bandwidth constraint is quantized. Each iteration the *MC-DCP* solution finds the path with the maximum probability that accomplishes the delay constraint, and with the bandwidth constraint probability between two quantized values. Finally a utility function selects one of the nondominated paths.

In [89] Anjali and Scoglio propose an algorithm for traffic flow routing in a *MPLS* network managed by a Traffic Engineering Automated Manager (*TEAM*) [90]. This algorithm bases its decision on select the path minimizing a cost function. This cost is attributed to five factors: bandwidth requested, switching and signalling in relay (intermediate nodes), remaining available bandwidth and delay. According to the authors, in *MPLS* networks the routing research has been developed on Label Switching Path (*LSP*) routing, i. e. how to route the *LSPs* in the network. A scheme for traffic flows over the *LSPs* in *MPLS* networks had not been considered. A path is then defined as a concatenation of *LSPs*. Among the set of paths between a source node and a destination node, each path has a cost associated. The cost is computed for each *LSP* of the path taking into account the mentioned five factors (bandwidth requested, switching, signalling, remaining available bandwidth and delay) weighted each one by a weight factor, but the cost of a path is not just the sum of the costs of the *LSPs*. This is because the relay nodes between *LSPs* have to perform additional switching and signalling to the change in the encapsulation from one *LSP* to the other. Thus, in the final cost of a path there are additional weights to take into account, the *IP* switching and signalling due to the presence of relay nodes.

Once the costs of all the paths between a source and destination are computed an algorithm has to select one of the paths. This selection tries to achieve a balance between maximize available bandwidth and minimizing the number of hops and delay. The algorithm considers a maximum number of F paths, from the path of only one *LSP* and then considering paths of 2 *LSPs* and so on. These paths are found without any consideration of feasibility. Next, the paths are checked for feasibility constraints, that is a minimum available bandwidth, a maximum number of hops and a maximum delay. The set of feasible paths among the F previously computed will be the possible candidates. The path with least cost is then selected. This proposed algorithm does not take into account the information of inaccuracy of the network state. However, the algorithm with some modifications can operate with inaccurate network state information. The proposal is to use a different algorithm to estimate and forecast more accurate information about bandwidth and delay of the *LSPs*. In any instant of time in the middle of the update period the bandwidth or delay of a *LSP* can be forecasted from the past P updated samples. Moreover another algorithm adjusts the value P, number of samples considered, based on the forecast performance.

Yia et al in [91][92] propose a different approach to deal with the inaccuracy of the network state information. In this strategy the connection requests with specific request demands are assigned to one or several alternative paths previously computed. In the source nodes the paths are precomputed periodically (not reacting to an incoming request), with the period equal to the interval between two consecutives updates. These precomputed routes are either the K-shortest or the K-widest depending of the algorithm. They are computed from the source to all the destinations in a graph where the links with a residual bandwidth lower than what a typical call are pruned. When a connection request demands a connection between the source and the destination nodes, a path is selected among the previously computed paths. Authors propose 5 algorithms to select the path. The first is BKW (Best-K-Widest), it selects the path whose bottleneck bandwidth most tightly fits the requested bandwidth among the set of K-widest paths. The second is RWK (Random-Widest-K), it selects randomly a path among the set of K-widest paths. The third is SWK (Shortest-K-Widest), it selects the shortest path among the set of K-widest paths. The fourth is BKS (Best-K-Shortest) it selects the path whose bottleneck link most tightly fits the connection request bandwidth among the set of K-shortest paths. And finally, the WKS (Widest-K-Shortest), it selects the path with largest bandwidth in its bottleneck link among the set of K-shortest paths.

In [93] Rétvari et al propose a precomputation scheme based on the *MIRA* (Minimum Interference Routing Algorithm). *MIRA* was originally designed with the assumption that the routing algorithm utilizes accurate information about the availability of unreserved bandwidth in the links of the network, to compute the critically of these links. The main contribution of the proposal in [93] is a novel characterization of the link critically, the critically threshold, that deals with both the complexity computation of *MIRA* and the routing inaccuracy problem. Based on the new critically scheme authors define a new routing algorithm, the Least-Critical Path First algorithm.

Since link critically reflects the network state that was valid the last time when a state of the network state information occurred, it is not necessary to be computed for every new connection request, only for every new network state update. The critically precomputation scheme was yet proposed in [77] doing it more realistic and with less computational complexity. However, the *MIRA* scheme with these modifications exhibits 'poor' performance. Authors of [93] propose some modifications to improve the *MIRA* precomputation scheme. They observe that it exists a well defined threshold of the capacity of any link, such that if the available bandwidth falls beyond this threshold then the link turns to critical This threshold value is computed as follows. First, it is calculated the maxflow value for a source-destination pair when the capacity of that link is set to infinite. Then it is calculated the maxflow for the same source-destination pair when the capacity of that link is set to zero. The difference between these two maxflow defines the critically threshold of that link for that source-destination pair. On the other hand for every link of the graph it is computed a committed load. This committed load of a link is a weighted sum of the critical threshold value of that link for the different source-destination pairs. All the links in the graph will have a cost assigned; this cost is computed depending on the committed load of the link, the requested bandwidth and the available bandwidth of the link. Finally, the least-critical-path-first algorithm selects the shortest weighted path.

11. The Predictive Approach in Routing

In this section it is reviewed some of the previous work related to self-learning routing or predictive routing. That is, mechanisms that learn which is the best route from the information obtained from the previous behaviour or performance. These schemes can include methods that predict the future traffic load on the links of the network, or works that base their routing decisions on the past blocking rate performance.

11.1. Hot-potato Routing

One of the older predictive (or self-learning) methods in network systems is the well-known hot potato routing scheme [94] proposed by Baran, which 'predicts' the best route to a destination node based on the delay information coming from that node. This scheme is applied to hop-by-hop routing. Every node has no buffers to store the information in transit and it only selects the next node to forward immediately the information. The hot-potato routing does not require any explicit flow control. A flow control is any kind of mechanism that inhibits to route a traffic request over a path even when it can, for example waiting an acknowledgement before send the traffic, or negotiate network bandwidth. Routing without flow control reduces the signalling overhead and also it is especially useful for situations where not all nodes generate packets at the same rate.

In hot-potato routing, it is assumed that the delay information is the length of the path (number of hops) and also that links are bidirectional. Every message of information (traffic request) contains a field which is set to zero upon initial transmission of the message. Every time a message passes through a node the value of this field is increased. When finally the message arrives at the destination node this field contains the length, in number of hops, of the path that this messages has traversed. With this information the nodes can build a routing table. In this table, for every destination node and for every possible output link there is pointed out a length value. This length value indicates the length of the path from that destination node when a message has arrived through that link. If a new message has to be send to any destination node, the output link selected will be that with the lower length value. If this link is busy the link with the next best length value is selected. Initially the routing table is set with high values. These values are updated depending on the traffic received and after a certain period the table will contain the path length to each of the destination nodes.

In [95], a dynamic variant of hot potato routing is presented by Busch et al The algorithm proposed is greedy, that is, a message or packet always tries to follow any good link. A good link is one that brings the packet closer to its destination and a bad link does not. If one packet can not follow any good link because they are occupied by other packets, it is forced to follow some bad link (it is deflected). In order to resolve conflicts when two or more packets are competing for the same output link, the algorithm uses priorities. The packets of a higher priority are routed before. To implement the priority the packets have states, where each state corresponds to a priority. The possible states are: sleeping, active, excited or running. Running is the state with highest priority and sleeping and active the lowest.

Initially when the packet is injected into the network it is in a sleeping state. The sleeping packet tries to become an active packet with a probability inversely proportional to the number of nodes of the network. Otherwise it remains at the sleeping state. Once the packet leaves the sleeping state it never returns to this state. The sleeping and active packets follow any good link if it is possible, otherwise they are deflected.

When an active packet is deflected because all the 'good links' are occupied by packets with higher priority, it has a chance to increase its priority and become excited with a certain probability also inversely proportional to the number of nodes. An excited packet tries to follow the link towards the home run path. The network simulated is an *nxn* mesh network, and then the home run path follows first the row path towards the destination and then the column path to the destination. When an excited packet successfully reaches the first node of the home run path it becomes a running packet. If an excited or running packet can not follow, for conflicts with other packets, the link toward the home run path it loses it priority and becomes active.

11.2. Estimation of the Link Available Bandwidth

In [96] Anjali et al propose to predict the future traffic load in a link through past measured samples of the traffic load in that link. The estimation algorithm predicts the available bandwidth in a link and also tells the duration for which the estimate is valid.

When a new bandwidth request arrives at a source node the bandwidth estimation algorithm is run for the links that do not have an available estimation of bandwidth. This estimation of the bandwidth on a link is formulated as a linear prediction with prediction coefficients that utilizes a certain number of past samples of the available bandwidth on the link. Based on the traffic dynamics, the number of past samples needed and the number of future samples predicted are changed. The problem can be solved using Wiener-Hopf equations. Once the available bandwidth of the links is estimated, the path selection algorithm computes the shortest widest path algorithm. If the bandwidth requested is larger than the bandwidth on the bottleneck link multiplied by a threshold value, the path is rejected, otherwise it is selected. If the path is not selected the next shortest widest path is computed, and so on. The parameter of threshold is used as a benchmark for path selection. If the bandwidth requested is more than a certain fraction of the bottleneck link, the request is rejected in this path. This is done to limit the congestion in the network

11.3. Proportional Routing

A different proposal of Nelakuditi et al is the 'proportional routing', proposed in [97] and [98] where the routes are selected without taking into account network state information. Authors in [97] focus on localized *QoS* routing schemes where the source nodes use only "local information" and thus it is reduced the signalling overhead associated on flooding the network state information. In localized approach for *QoS* routing, no global *QoS* state information exchange among network nodes is needed. Instead, source nodes infer the network *QoS* state based on flow blocking statistics collected locally, and perform path selection using this localized of the network *QoS* state. The algorithm inferred from this mechanism is the Proportional Sticky Routing (*psr*). In this algorithm, each source node has defined a set of candidate paths to each of the destination nodes. When a connection is requested in any source node, the *psr* algorithm selects a path among this set of candidate paths based on flow blocking probability. The *psr* scheme operates in two phases: proportional flow routing and computation flow proportions.

The proportional flow routing proceeds in cycles of variable length. During each cycle incoming flows are routed along the paths of the set of predefined candidate paths. A path is selected depending on a frequency defined by a proportion. A number of cycles forms an observation period, at the end of which a new flow proportion for each path is computed based on its observed blocking probability. The set of predefined candidate paths include the shortest paths (minimum number of hops) and also alternative (and longer paths). Without explicit trunk reservation the scheme includes the self-refrained alternative routing, to route preferably along the set of shortest paths. This is achieved because an alternative path is only selected if it has 'better quality' than any of the shortest paths. The quality of the paths is measured in terms of blocking probability.

The *psr* algorithm runs as follows. Each of the predefined paths (shortest and alternative) between a source and destination has a maximum permissible flow blocking parameter and a flow blocking counter. This maximum permissible flow blocking parameter is the number of times a flow is blocked when selecting that path before the path is considered no-eligible. At the beginning of each cycle the flow blocking counter is set to the value of the maximum permissible flow blocking. Every time a flow routed along a path is blocked, its flow blocking counter is decreased. When the counter reaches zero, the path is not considered in the path selection (no-eligible path).

Among the set of eligible paths (with flow blocking counters different from zero), the selection is done used a weighted-round-robin path selector (*wrrps*) also defined in [97]. The *wrrps* is implemented by using a deterministic sequence of paths which have the property that the paths are distributed periodically with a frequency which closely approximated the prescribed flow proportions. Every time the eligible set of paths changes, a new sequence of paths is generated.

When the set of eligible paths becomes empty, the current cycle finishes and a new cycle starts. When a cycle is started the set of eligible paths is all the set of predefined candidate paths, and the flow blocking probability counters of these paths are set to the maximum permissible flow blocking of each path.

Concerning to the second phase, computation flow proportions, these are computed at the end of each observation period. The number of cycles that forms an observation period is also a configurable parameter. During each observation period the number of flows routed along each path is counted. Since the maximum permissible flow blocking parameter is the same for all the shortest paths, it is demonstrated that with this method at the end of the period all the paths have the same blocking rate. On the other hand for alternative paths the maximum permissible flow blocking parameter is adjustable between 1 and the value of the maximum permissible flow blocking parameter of the shortest paths. The minimum blocking probability among all the shortest paths is computed at the end of the observation period. This minimum probability is used as a reference to control the flow proportions of the alternative paths. If the blocking probability of an alternative path computed at the end of the period is larger than the minimum probability of the shortest paths, then the maximum permissible flow blocking parameter of this alternative path is decreased. Instead, if the blocking probability of the alternative path is lower than a certain percentage of the minimum probability of the shortest paths the maximum permissible flow blocking parameter is increased. Otherwise, the blocking probability of the alternative path is between a percentage of the minimum blocking probability and the minimum blocking probability, the maximum permissible flow blocking parameter of this alternative path is not changed. This percentage of the minimum blocking probability of the shortest paths is also a configurable parameter to limit the 'knock-on" effect.

In [98] the same authors propose a hybrid method that uses both local and global information, and hence requiring network state information updates. The global information about the network state is used to select the set of candidate paths to ensure that the localized scheme adapts to varying network conditions. The paths selected as candidate paths are the widest link disjoint paths (*wlp*), and they are updated based on the global state network information. Basically, the candidate paths must not share its bottleneck links. The 'width' of a set of path is computed, that essentially accounts for the sharing of the links between paths. A new path is included in the set only if it decreases the width of the set.

However, the local information is used to route the flows by means of proportional routing. The main advantage comparing it with other mechanisms that need updates of the network state information is that these updates can be infrequent, and then the signalling overhead is smaller.

11.4. Other Predictions

In [99] Foag et al present a traffic prediction algorithm for speculative network processors. In network processors, due to the sequential packet-layer processing, processing delay is not optimized. To solve this problem, speculative packet processing is applied, which requires accurate traffic prediction. The concept of speculative packet processing comprises two components: protocol stack prediction and speculative data processing.

To decrease the processing latency the protocol stack prediction has to provide high accuracy. The speculative data processing solves the network layer dependences.

The decision of how to handle and forward a packet is done upon the information which is contained in the packet header. The header of the data-link layer, network layer and transport layer are hierarchically arranged. And without prediction the processing of the packet header of each layer has to be done in a serial way. To speed-up the execution of the network processor, the protocol stacks predictor predicts the protocol stack of the next packet, from the history of packets received earlier.

From this prediction the data dependence between hierarchical levels is solved and the processing of headers of data-link layer and transport layer can be done simultaneously. That is, from the prediction result the packet processing starts speculatively and the whole process is speeded.

When the task of extracting the headers of each level is really done, the prediction can be verified. In case of misprediction results have to be dropped and the processing has to be restarted.

The different protocol stacks supported by the system are defined by a Stack Identifier (*SI*). It contains a unique data pattern for each stack. Output of the prediction is the SI of the packet which is expected to be received.

The prediction is based on the history of packets received earlier. The protocol stack predictor predicts the *SI* of layer 3 and layer 4 from the *SI* of layer 2 extracted from the packet; and the packet processing is started simultaneously. When the *SI* of packet 3 is really computed the prediction is verified. If it is incorrect a new prediction of *SI* of layer 4 is needed from the real value of *SI* of layer 3. Otherwise no more prediction is needed. When *SI* of layer 4 is really computed the prediction of layer 4 is verified. In any case of misprediciton (layer 3 or layer 4) the process has to be aborted and restarted.

The computation of the predicted *SI* values follows the policy of most frequently used. As an extension of the policy an additional weight factor is appended during prediction to each *SI*. The objective is to prioritize delay sensitive traffic in case of equal stack frequency. The prediction tables that record the most frequently used *SI* value, are named Decode History Tables (*DHT*). To index and access one entry of the *DHT* table is needed a *PSSW* (Protocol-Stack-Status Word). The *PSSW* contains the type of layer 2, 3 and 4; and two flags that indicate if *SI* of layer 3 and/or layer 4 has been yet really computed.

A different work in the field of prediction in networks is presented by Kim et al in [100]. It is proposed a routing mechanism through the least cost delay constraint based on prediction of the average queuing delay. They analyze the packet delays of a single server queuing system with self-similarity traffic. The average queuing delay is a major contribution to the packet delay due to the fact that the queuing delay rapidly increases as the utilization of the router increases.

The average queuing delay with self-similarity traffic can be computed through an analytic model which is based on queuing theory. This involves some complex computation but by applying polynomial approximations of the 3rd degree the computing is simplified. This is applicable to the various ranges of the Hurst parameter, H. The authors propose to predict the average queuing delay from different measured H parameters. They also propose a cost function that can be used to route through the least cost delay by using the predicted average queuing delay.

12. The *PBR*-Mechanism in *IP/MPLS* Networks

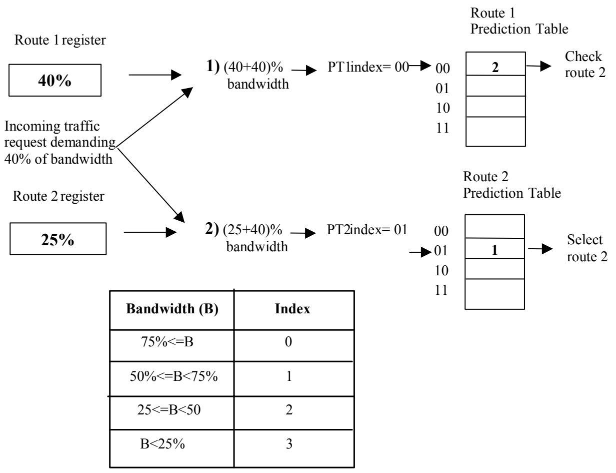
12.1. Motivation.

Unlike the previously mentioned predictive schemes, whose objective is to predict the incoming traffic load or inferring the best path from the blocking statistics, the *PBR* applied in *IP/MPLS* networks, focuses on predicting link and route availability. Moreover, the *PBR* mechanism also significantly reduces the signalling overhead due to the fact that update messages are not required. Similar to the 'proportional routing', in the *PBR* routes are selected without taking into account network state information coming from update messages. However, in 'proportional routing' the route selection is based on flow blocking statistics collected locally, whereas in the *PBR* the route is predicted to be blocked or not based on both, Prediction Tables, and local information.

12.2. Description and Data Structures.

The first approach to the *PBR* mechanism for *IP/MPLS* networks was presented in [101]; and it is based on choosing the possible routes between different fixed alternate routes. In this work the route is chosen between 2 (k in general) static (fixed) and previously computed routes. The main reason motivating the use of fixed precomputed routes is to limit the number of Prediction Tables in the sources nodes; using fixed alternate routes the number of Prediction Tables is limited. Later, new algorithms inferred from the *PBR* that select among more routes were proposed in [102].

Unlike branch prediction where the history of prediction outcomes is stored in a register, in a network scenario it is necessary to keep the network state from the point of view of the source node. In order to achieve it, the *PBR* mechanism registers the amount of bandwidth that every source node allocates to every route originated on such a source node. For simplicity of exposition, it is assumed that the information about both available and used bandwidth is expressed in terms of a percentage of the total capacity of the end-to-end route. There is one register per route on every source node. These route registers are updated with information about assigned bandwidth from the point of view of these source nodes. One of the main characteristics of the *PBR* mechanism is that the register's updating



Bandwidth (B)	Index
75%<=B	0
50%<=B<75%	1
25<=B<50	2
B<25%	3

Figure 39. *2-PSR_FA* performance, bandwidth codified with 2 bits.

process is achieved without distributing update messages. Because of the removal of these update messages, the bandwidth allocated in the route registers of the source nodes does not reflect the precise bandwidth assignment values.

The information about assigned bandwidth is used to access the Prediction Tables; hence it should be digitalized in order to constitute a proper table index. As an example, if a single bit is employed for digitalizing the bandwidth information, it is possible to assign ‘0’ to the index when the used bandwidth in the route is larger than or equal to 50%, otherwise a ‘1’ is assigned. Table in Figure 39 shows the index values for two bits.

Source nodes include one Prediction Table for every feasible route. Every route register has its corresponding *PT*. The *PTs* have different entries, each keeping the information about a different pattern by means of a two-bit counter. The use of two values to account for the availability or the unavailability has been widely studied in the area of branch prediction in computer architecture.

The number of entries of the prediction tables depends on the number of bits of the route registers. For example, if route registers keep information about the used bandwidth in the route within two bits, then the number of entries of the Prediction Tables is 4.

12.3. Off-demand algorithm inferred from the *PBR* Mechanism.

Based on the *PBR* off-demand mechanism, it is proposed the *k-PSR_FA* (Predictive Selection of Route Fixed Alternate) algorithm, being k the number of feasible routes [102]. Figure 39 illustrates an execution of the algorithm. In the example of Figure 39, it is assumed that there are two precomputed shortest routes between every source-destination nodes pair, and that the assigned bandwidth is codified by two bits. Figure 39 depicts the handling of a new request that demands 40% of bandwidth. It is also assumed that these shortest routes are link disjoint, if possible. Otherwise the shortest routes should share the minimum number of links. This is done because if the first route is predicted to be blocked, then the prediction is effectively to use a completely different route, since the source node does not know the identity of the link blocking the first route. Generally, the *k-PSR_FA* algorithm checks the k-shortest routes in a computed order, according to the availability of their links. The information about the availability of the links does not represent the current picture of the network. Indeed, without updating, every node only knows how routes and links have been used in the past. This information dictates the order by which the *PTs* are checked. Getting back to Figure 39, the last information upon the first route is a used bandwidth of 40%. This used bandwidth is incremented by the requested bandwidth, i.e. 40%+40%. If the resulting figure is lower than 100 %, then the *PT* of the first route is checked, that is the counter of the corresponding entry is read; otherwise, the next *PT* would be checked. In the example, the total bandwidth is 80% (>75%), so that the index used to access the first *PT* is 00. With this index, the *PT* of the first route is accessed and the counter is read. According to Figure 39, the value obtained after accessing the *PT* is 2, hence the decision made by the prediction process is to avoid the first route. Hence, the second route is examined. In this second route, the used bandwidth is 25%, so that the resulting figure is 40%+25%=65%. This means an index of 01. The *PT* of the second route is accessed with this index, obtaining a value of 1. According to this counter value, the algorithm selects this second route. It is necessary to point out that the algorithm checks both the counter value of the *PT* and the availability of the node's output links towards each of the two routes, as nodes always have updated information on the availability of their output links.

In Figure 40 it is presented a short summary of the *k-PSR_FA* algorithm, for k=2. The functions that check the availability of route 1 and route 2 are called as Check(Route1), and Check(Route2), respectively. In the example, after checking the *PTs* of both routes, if the algorithm still has not selected any route according to the prediction, the algorithm will select the route by only checking the availability of the node's output links. These functions are termed CheckF(Route1) and CheckF(Route2), respectively.

```
New request demanding an X% of bandwidth.
Check(Route 1):

    The new bandwidth is added to the bandwidth kept in the route1
    register (Y%). The total bandwidth is  X+Y%.

        If (X+Y)% <=100% the PT of the first route is checked
            If(PT counter<2) and there is availability in the output
            link the algorithm selects the  route1
            Else Check(Route 2).
        Else Check(Route 2)

Check(Route 2) :

    The new bandwidth is added to the bandwidth kept in the route2
    register (Z%). The total bandwidth is X+Z%.

        If (X+Z)% <=100% the PT of the second route is checked
            If (PTcounter<2) ) and there is availability in the output
            link the algorithm selects the  route2
            Else CheckF(Route 1)
        Else CheckF(Route 1)

CheckF(Route 1):

    The new bandwidth is added to the bandwidth kept in the route1
    register (Y%). The total bandwidth will be X+Y%.

        If (X+Y)% <=100%
            If there is availability in the output link the algorithm
            selects the  route1
            Else CheckF(Route 2).
        Else CheckF(Route 2)

CheckF(Route 2):

    The new bandwidth is added to the bandwidth kept in the route1
    register (Z%). The total bandwidth will be X+Z%.

        If (X+Z)% <=100%
            If there is availability in the output link the algorithm
            selects the  route2
            Else No route is assigned
        Else No route is assigned
```

Figure 40. Summarizing the *2-PSR_FA* algorithm.

The route registers at the source node are updated with the information about the used bandwidth for the source node in every route. In the example above, when the algorithm selects the second route, the new bandwidth used by this node in this second route will be 65%. It is important to note that this used bandwidth is just the value known by the node, which might be substantially different from the real bandwidth occupation. This is because, due to the lack of update messages, bandwidth changes produced by other source nodes allocating bandwidth on links of the same route are not reported.

An important issue to be considered is that only the *PT* of the selected route is actually updated (or trained). Hence, if the connection is established, the corresponding counter on the *PT* is decreased, otherwise (i.e., the connection is blocked) the counter is increased. In the example, if the connection is successfully established, the counter of the entry 01 in the *PT* of route 2 will be 0, but if the connection is finally blocked the counter will be 2. The attempt of selecting the route by just checking the output availability when no route is assigned is done to unblock the *PT* counters. Indeed, if the route is selected and the connection can be established, then the corresponding *PT* counter of route 1 or route 2 is decreased, hence unblocking it.

12.4. On-demand algorithm inferred from the *PBR* Mechanism.

It is necessary to clarify what it is assumed 'for on-demand' algorithm. In general an 'on-demand' algorithm can compute dynamically the possible route among all the possible routes between a source and a destination node. But in the particular case of the *PBR* mechanism it is necessary a *PT* for every possible route in the source nodes. This implies that all routes have to be precomputed and known to create their *PTs*. Hence, it is assumed that for every source-destination node pair, the source nodes calculate all the possible routes and create a Routing Table, with the possible routes ordered from the shortest to the longer in number of hops. Despite there is not updating of the network state information, it is necessary an updating of the network topology. Every time there is a change in the network topology it is necessary some type of update message flooded out through the network. When the source nodes receive these network topology messages they recalculate their Routing Table. For every route in these Routing Tables there is its corresponding *PT*. Note that, changes in network topology are more infrequent than changes in network state (load). Then, the 'on-demand' algorithm inferred from the *PBR* mechanism will select dynamically the route among the previously precomputed. Taking account this characteristic, the difference between the previously proposed 'off-demand' *k-PSR-FA* algorithm and the possible 'on-demand' algorithm inferred from the *PBR* mechanism is tenuous. However, the objective of proposing the 'on-demand' algorithm is to design an algorithm from the *PBR* mechanism able to manage more routes, even all the possible routes between a source and a destination node. Moreover, in the results presented in the performance evaluation of these algorithm, it is assumed that in the 'off-demand' *k_PSR-FA* algorithm the possible routes are precomputed manually and are either link-disjoint or sharing the minimum number of links. But in the 'on-demand' algorithm, all possible routes are computed by the algorithm creating a Routing Table.

As exposed earlier, the potential problem of a *PBR* on-demand mechanism is the amount of memory required by both the number of *PTs* and the size of the *PTs*. Remember that source nodes include a *PT* for every possible route to every possible destination. In addition, a large number of *PTs* negatively impacts on the computational cost.

The problem of the memory requirements has been addressed by means of both, reducing the PT size, and reducing the number of *PTs*. First, the *PT* size is reduced so that there is only one entry of two bit counter in every *PT*. As a consequence, it is not necessary to codify the requested bandwidth in a certain number of bits, since the algorithm does not consider it in the route selection (because there is only one entry of one two-bit counter on each *PT*). For every new connection request the corresponding *PTs* of the possible routes are accessed and read, independently of the requested bandwidth. This is done to both, limit the necessary amount of memory required, and simplify the execution of the algorithm. Second, the algorithm is able to calculate all the possible routes and then check all the possible *PTs*. However, to reduce even more the memory requirements, a new parameter, R, is added. R is the number of statically precomputed shortest routes. Then, in the source nodes, there are R *PTs* for every source-destination pair of nodes. The algorithm would create the Routing Table with the R shortest routes in number of hops. In each entry of the Routing Table there would be a field with the corresponding *PT* (of only a two-bit counter) of the route.

Despite the fact that the number of *PTs* has been reduced as well as their size, a significant computational cost is needed to access all the feasible *PTs*. Hence, to reduce this computational cost the number of routes to be compared is limited adding a new parameter k. k is the dynamically k-shortest routes with two-bit counter lower than 2 and with output link availability. The routing algorithm inferred from the *PBR* on-demand mechanism is named Predictive Selection of Route on Demand (*R-PSR_k*) [102]. In short, the *R-PSR_k* algorithm checks the k-shortest routes with two-bit counters lower than 2 and with output link availability among the first R shortest routes, in number of hops.

```
For(i=1 to R)  (R can be=all possible routes){

      While(CheckedRoutes<=k){

      If(two-bit_counter(Route(i)<2) and there is output link availability{
      CheckedRoutes++;
          If(Length(Route(i)<Length(AssignedRoute)) AssignedRoute=Route(i);
            If(Length(Route(i)==Length(AssignedRoute)){
                CheckLocalLinkAvailability:
                If(LocalLinkAvailability(Route(i))> LocalLinkAvailability(AssignedRoute))AssignedRoute=Route(i);
            }Endif
          }Endif

      }Endwhile

}Endfor

If any route is assigned run the same algorithm without checking two-bit_counter values:

For(i=1 to R)  (R can be=all possible routes){

      While(CheckedRoutes<=k){

       If  there is output link availability{
       CheckedRoutes++;
          If(Length(Route(i)<Length(AssignedRoute)) AssignedRoute=Route(i);
           If(Length(Route(i)==Length(AssignedRoute)){
              CheckLocalLinkAvailability:
              If(LocalLinkAvailability(Route(i))> LocalLinkAvailability(AssignedRoute)) AssignedRoute=Route(i);
            }Endif
          }Endif

      }Endwhile

}Endfor
```

Figure 41. Summarizing the *R-PSR_k* algorithm.

Once the problem of the memory requirements has been fixed, the *R-PSR_k* algorithm is described. The *R-PSR_k* algorithm looks, for every new connection request, the possible routes and reads the two-bit counter values as follows. Once the routes are calculated they are checked according to their length in number of hops. The first, shortest route is checked. If its corresponding two-bit counter is lower than 2 and the corresponding output link has enough available bandwidth the route is provisionally selected. In any case, if the first route is selected or if it is not selected, the next, second route, is checked. If the second route has its two-bit counter lower than 2, the same hop length than the first and output link availability, this second route is compared with the first. If the second route has more available bandwidth, this second route is now provisionally selected. This process finishes when k possible routes are considered (k shortest routes with two-bit counter lower than 2 and output link availability) or when R routes are checked. See in Figure 41 a summary of this *R-PSR_k* algorithm. In order to make understanding easier the *R-PSR_k* algorithm can be compared with the Widest Shortest Path (*WSP*) [75]. The *R-PSR_k* algorithm runs similar than the *WSP* but with two differences. The first is that the algorithm selects the widest shortest route between the routes with counter lower than 2 and output link availability. That is, it selects the widest shortest route in a graph where the routes with two-bit counters larger than 1 or no output link availability are pruned. The second difference is that *R-PSR_k* uses the local information on the source node about the link availability of the routes. This local information stands for the amount of bandwidth allocated by those connections originated by such a source node.

As in the *k-PSR_FA* algorithm, if the *R-PSR_k* algorithm does not select any route, the routes are checked as explained above but eliminating the restriction of two-bit counters lower than 2. See also summary in Figure 41.

The *R-PSR_k* algorithm updates (or trains) the two-bit counters of the *PTs* according to the following. If the connection can be established the two-bit counter corresponding to that route is decreased, otherwise, the connection is blocked, the two-bit counter is increased.

12.5. Performance Evaluation.

In order to evaluate our proposal the performance of the *PBR* mechanism is compared with a well-known *QoS* routing algorithm, the Widest Shortest Path (*WSP*). For every new incoming request, the *WSP* dynamically selects the route with the largest amount of available bandwidth among the shortest (i.e., minimum-hop) ones. All the performed simulations are obtained by applying both *PSR* algorithms and the *WSP* algorithm on the *KL* topology [77], depicted in Figure 42. In these simulations nodes 1, 2, 11, 12, 14 and 15 in Figure 42 act as source and destinations nodes. Connection arrivals are assumed to be Poisson, and all the links have the same available bandwidth, which is normalized to 100%. Each arriving connection requires a certain percentage of the total bandwidth. The holding and arrival times of the incoming requests are measured in units of time. All the connection requests have an average holding time of 10 units and an average arrival time of 10 units. In order to change the traffic load, the average requested bandwidth (that is, the average value of all the requested bandwidths) demanded by the incoming requests ranges from 10% to

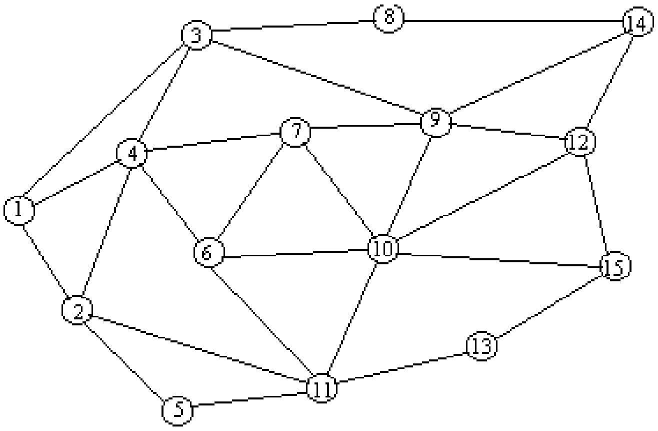


Figure 42. KL Topology used in the simulations.

25 %.Three set of simulations are carried out. The first set of simulations targets to find out the optimal number of bits needed to codify the bandwidth requirements in the *k-PSR_FA* algorithm. The second targets to evaluate the *PSR* performance, compared to with the *WSP* algorithm. And finally the impact of the parameters R and k on the *R-PSR_k* algorithm performance is evaluated on the third set of simulations. In all the simulations 15 000 connection requests are simulated.

12.5.1. Number of bits to codify the requested bandwidth

As it is exposed in section 11.3, the *k-PSR_FA* algorithm uses the bandwidth codification in the process of selection of the route. In this first set of simulations the impact on the *k-PSR_FA* performance is evaluated when the number of bits used to codify the bandwidth changes. Notice that the length of the route registers and the number of *PT* entries depend on the number of bits used to codify the bandwidth. For example if the number of bits used to codify the bandwidth is 3, the route registers will have a length of 3 bits, and the *PTs* will have 8 entries each one, but if the number of bits is 0 (bandwidth is not codified) there will not be route registers and the *PTs* will have only one entry. In these simulations the two routes precomputed for the *2-PSR_FA* algorithm are the two shortest and link disjoint; and for *4-PSR_FA*, the first 3 routes are the shortest and link disjoint, while the fourth shares the minimum number of links with the other 3, because there are not 4 link disjoint routes in the topology simulated. In Table 8 there is represented the percentage of blocked connections, for the *4-PSR_FA* algorithm for 0 (bandwidth is not codified), 1, 2 and 3 bits to codify the requested bandwidth, and for 10%, 15%, 20% and 25% of average requested bandwidth. For 10%, 15% and 20% the best results are for 0 bits; only for 25% the best results are for 2 bits. Similar results are obtained for *2-PSR_FA*. On average, for our range of traffic load the best results are usually for 0 and 2 bits. For simplicity and taking into account that 0 bits implies that there are not route registers, and only one *PT* of one two-bit counter per route is required in the source nodes, in the rest of the performance evaluation only results for 0 bits are presented for the *k-PSR_FA* algorithm.

Table 8: *4-PSR _FA* % of blocked connections vs. the number of bits to codify the requested bandwidth.

Average	**Number of Bits**			
Requested Bandwidth	**0**	**1**	**2**	**3**
10%	0,3314%	0,3314%	0,3321%	0,40262%
15%	1,2682%	1,5041%	1,3434%	1,6959%
20%	3,9550%	4,8036%	5,5262%	5,2469%
25%	12,1375%	11,8983%	11,2713%	12,9306%

12.5.2. Blocking Probability versus Traffic Load.

The two *PSR* algorithms, *k-PSR_FA* and *R-PSR_k*, are compared with the *WSP* algorithm. Two *WSP* versions, *WSP* with off-demand route calculation, named *k-WSP_FA*, with k link-disjoint routes (if possible), and *WSP* with on-demand route calculation named *R-WSP_k*, are simulated too.

Figure 43 shows results of the percentage of blocked connections versus the time between updating (in units of time) for 10%, 15%, 20% and 25% of average requested bandwidth. In these simulations k=2 and k=4 are assumed for the *k-WSP_FA* and *k-PSR_FA* algorithms, and k=All and R=All for the *R-WSP_k* and *R-PSR_k* algorithms. In the off-demand algorithms that use precomputed routes (*k-WSP_FA* and *k-PSR_FA*) the routes have been manually selected. For 2-FA, the two routes are the two shortest link disjoint and for 4-FA, the first 3 routes are the shortest link disjoint, while the fourth shares the minimum number of links with the other 3 (because there are not 4 link disjoint routes in the topology simulated). Remember that the *PSR* algorithms do not vary their performance with the updating time because they do not need update messages.

From the obtained results, the conclusion is that both *PSR* algorithms outperform the *WSP* algorithms when the network state updating time is larger than 5 units of time, except

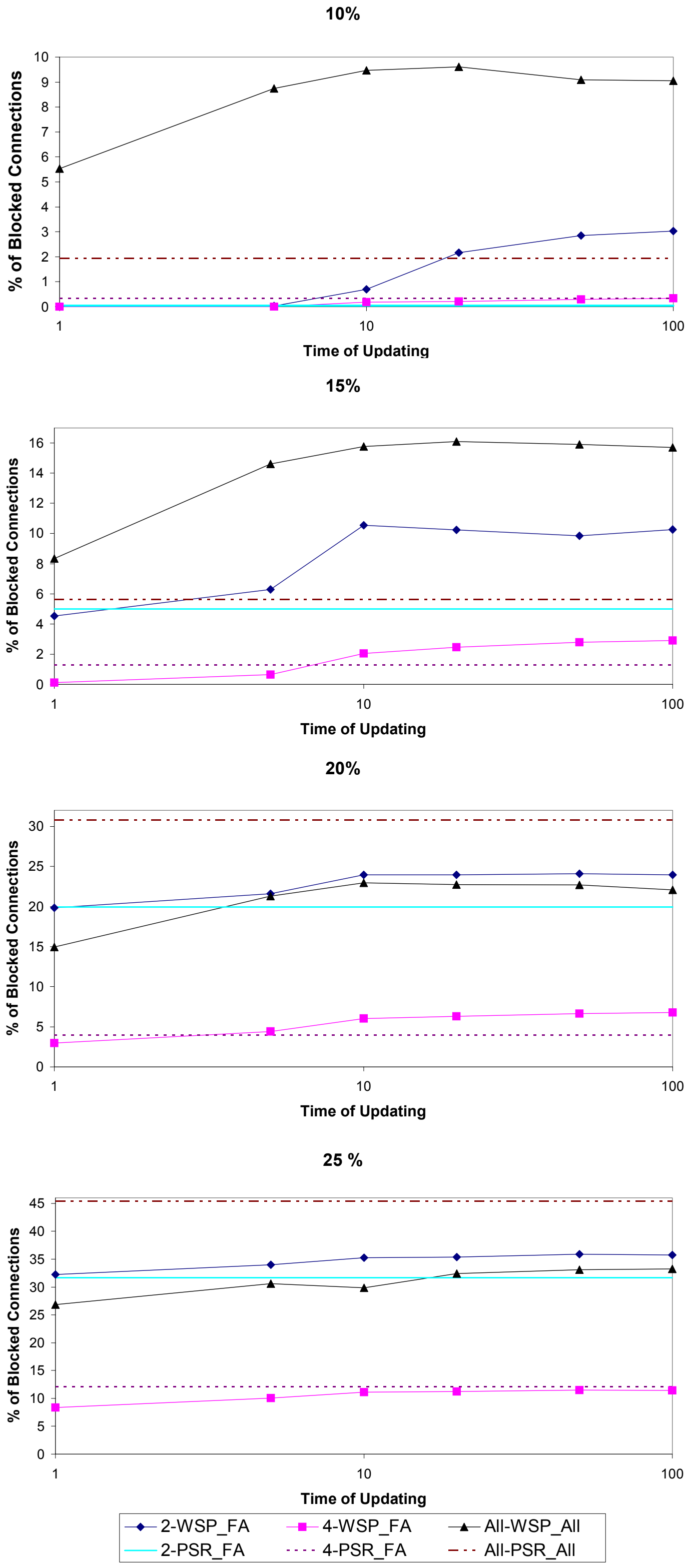


Figure 43. *PSR* versus *WSP* for traffic load of 10%, 15 %, 20% and 25% of average requested bandwidth.

for high traffic load of 25% of average requested bandwidth. This means that the *PSR* algorithm presents worse performance with high traffic load. But in some cases the *PSR* outperforms the *WSP* even when updating is every unit of time (see graphic for 10 and 20% of traffic load). On the other hand, the *4-PSR_FA* algorithm outperforms in most cases the *R-PSR_k* algorithm. This effect is also observable in the *WSP* algorithms. This can be explained because more routes to select does not always imply better performance as stated Mitra et al in [103]. The *4-PSR_FA* algorithm only selects among 4 routes, but these routes has been previously and manually selected, being link disjoint the first three and sharing the minimum number of links the fourth. And then, the selection of the fixed alternate routes is as important as the routing algorithm [98].

12.5.3. Effect of the number of possible routes on the *R-PSR_k* algorithm.

From the previous results, the performance of the *PSR* algorithm depends on the number of possible routes to be selected. In these simulations the impact of the parameters R and k on the *R-PSR_k* algorithm performance is evaluated. Table 9 shows results of the *R-PSR_k* algorithm being the R parameter, either all possible routes, 100, 10 or 4 routes; and being the k parameter either R, 4 or 2.

Table 9 : % of blocked connections of the R-PSR_k varying R and k.

For 10% of traffic load

R/k	All	4	2
All	1,94	0,93	1,22
100	1,94	0,93	1,22
10	1,54	0,93	1,22
4	1,00	1,00	1,21

For 15% of traffic load

R/k	All	4	2
All	5,64	7,41	5,47
100	5,64	7,41	5,47
10	6,92	7,99	5,40
4	6,25	6,25	5,03

For 20% of traffic load

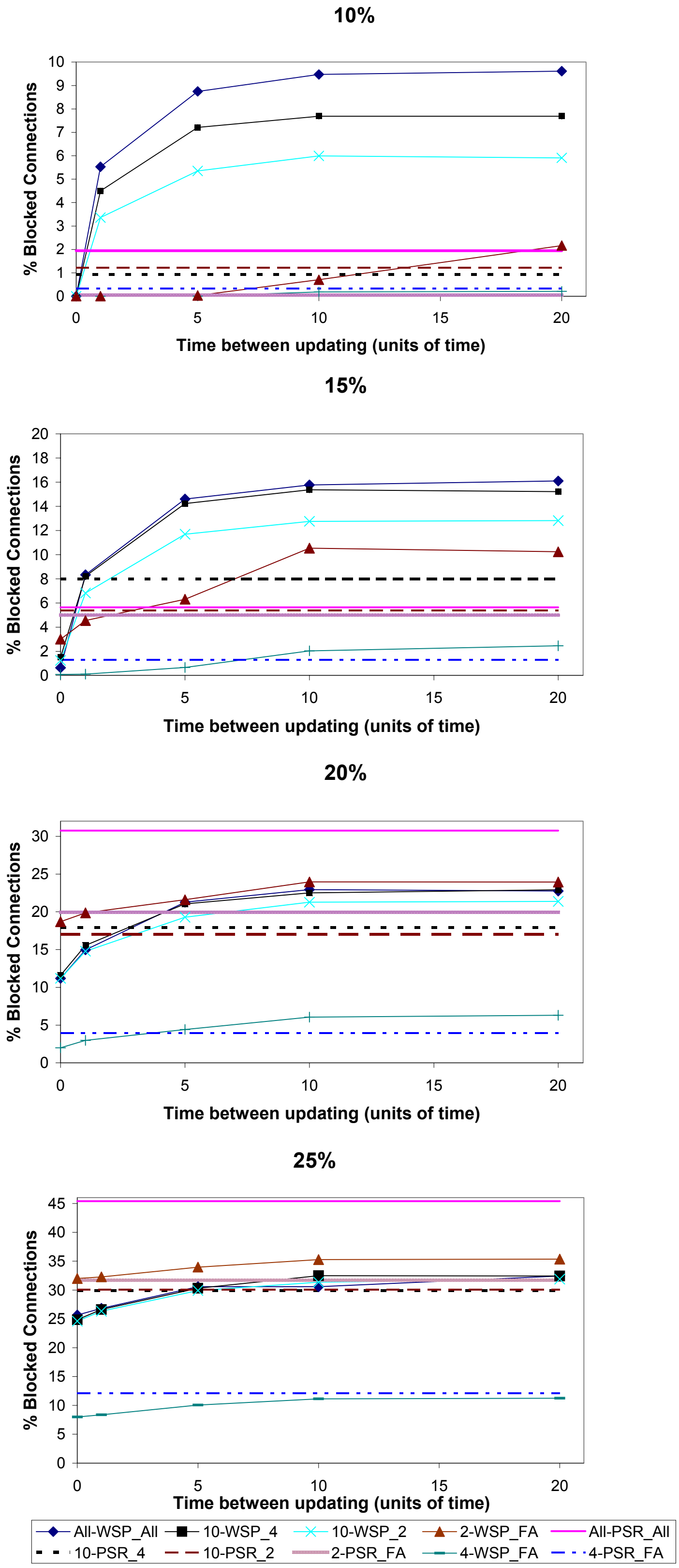
R/k	All	4	2
All	30,76	27,66	26,25
100	29,04	26,99	26,06
10	17,51	17,91	17,04
4	17,04	17,04	16,34

For 25% of traffic load

R/k	All	4	2
All	45,38	43,79	43,19
100	42,29	41,23	40,49
10	30,11	29,87	30,03
4	29,23	29,23	29,23

In general decreasing R the percentage of blocked connections decreases (except for 10% of traffic load and k=4). This effect is more significant for high traffic load, 20% and 25%. The reason is that R is the number of precomputed routes and hence of *PTs*. Remember that the *PTs* are trained by means of the blocked drives to a lower number of blocked connections. On the other hand when reducing the k parameter the percentage of blocked connections also decreases. This can be explained because there is an effect of trunk reservation. Just as an example, if for a connection request there are 4 possible routes of 3 hops with the two-bit counter lower than 2 and output link availability, the *4-PSR_4* algorithm will select the widest among these four (k is 4). Instead, if k is 2 the *4-PSR_2* algorithm will select only between the two first routes.

The next set of simulations targets to compare the *PSR* algorithm with the *WSP* algorithm, considering the k and R parameters. For coherence we also introduce the R and k parameters in the *WSP* algorithm. Despite the *WSP* algorithm does not utilize route registers, the number of routes to select is limited. The *R-WSP-k* selects the widest shortest route among the first k dynamically shortest routes with availability from the R statically shortest and stored in the Routing Table. In Figure 44 we present results of the percentage of blocked connections versus the time between updating for 10%, 15%, 20% and 25% of average requested bandwidth, comparing the *R-WSP_k*, the *R-PSR_k*, the *4-PSR_FA*, the *2-PSR_FA*, the *4-WSP_FA* and the *2-WSP_FA* algorithms. Note that updating every 0 units of


10%

15%

20%

25%

Figure 44. Effect of the R and k parameters on the percentage of blocked connections.

time means the ideal case where the *WSP* algorithm has always the entire network state information. For 10% of average requested bandwidth and updating ideal there are not blocked connections. This is due because the network has the resources necessary for the amount of traffic requested. For 15%, 20% and 25% with updating ideal the lowest percentage of blocked connections corresponds to the *All-WSP_All*, the *10-WSP_4*, the *10-WSP_2* and the *4-WSP_FA*. When the *WSP* algorithm has perfectly updated information it can optimally assign the requests among all the possible routes. But when the time between updating rises the percentage of blocked connections rises too due to the inaccuracy of the information used by the *WSP*. As in the *PSR* case if the updating is not ideal the *WSP* improves its performance when R and k decrease.

For 10% of requested bandwidth the best results of the *WSP* and the *PSR* algorithms correspond to the *4-WSP_FA* and the *2-PSR_FA* respectively. That is using 2 or 4 precomputed links disjoint routes or sharing the minimum number of links. We also observe that the *2-PSR_FA* algorithm outperforms the *4-WSP_FA* when updating is every 5 units of time. For 15% of requested bandwidth the best results among the *WSP* algorithms correspond to the *4-WSP_FA* algorithm; and for the *PSR* algorithms, the *4-PSR_FA* has the best performance. In general for 10% and 15% of requested bandwidth for both algorithm, *WSP* and *PSR*, the best performance appears when the algorithm selects among 4 shortest routes sharing the minimum number of links. Notice that good results are also obtained when the algorithms can select among the 2 shortest and link disjoint routes. And also for 10% and 15% of requested bandwidth the worst results are when the algorithms can select among all the possible routes between source and destination. That is, the more routes to select the worse performance. Moreover, when comparing the best results of both algorithms, the graphics of the *2-PSR_FA* (for 10%) or of the *4-PSR-FA* (for 15%) algorithms cross the *4-WSP_FA* graphic when updating is between 5 and 10 units of time.

On the other hand, for more traffic load (20% and 25% of average requested bandwidth) the best results for the *WSP* algorithms correspond to the *4-WSP_FA* algorithm and the second best results are for *R-WSP_k* with R=10 and k=2. Instead, the worst performance is for the *2-WSP_FA* algorithm. Notice that now selecting among 2 precomputed shortest and link disjoint routes produces the worst performance, but selecting among the 4 shortest routes sharing the minimum number of links produces the best performance. Also for the

PSR algorithms the best results are for the *4-PSR_FA* and the second best for the *R-PSR_k* with R=10 and k=2. But for the *PSR* the worst results always correspond when selecting among all possible routes, *All-PSR_All*. If we compare the best results of both, *PSR* and *WSP* for 20% of requested bandwidth, the *4-PSR_FA* algorithm outperforms the *4-WSP_FA* even when updating is every 5 units of time. However, also comparing the best results, for 25% of requested bandwidth the *4-PSR_FA* does not outperform the *4-WSP_FA* algorithm for updating from 0 to 20 units of time. The *PSR* algorithms degrade their performance with high traffic load more markedly than the *WSP*. This observation confirms previous results in Optical Networks for high traffic load and presented in the Part II of this Thesis.

12.5.4. Signalling Overhead.

The previous results show that for moderate traffic load the *PSR* algorithms outperforms an usual *QoS* routing algorithm, the *WSP*, when the network state information is updated every 5 units of time. In this section the signalling overhead produced by these update messages is evaluated.

The signalling overhead is evaluated by means of the number of update messages per unit of time, and the number of connection requests produced per update message. Notice that, when updating is ideal, the signalling overhead cannot be evaluated, because updating would be instantaneous. Table 10 shows the results of signalling overhead for the *WSP* algorithm when updating is every N units of time.

Table 10: Evaluation of the signalling overhead produced by the *WSP* algorithm.

N	**1**	**5**	**10**	**20**
# update messages/unit of time	1	0,20	0,10	0,05
# connection requests/# update message	3,01	15,09	30,03	60,07

The results in previous subsection have shown that only when updating every unit of time the *WSP* algorithms outperform the *PSR* algorithm. Table 10 shows that for the *WSP* algorithm when N=1 unit of time it would be necessary an update message for every 3 established connections. This implies that every 3 requested connections all the network state information has to be flooded and updated between the different source nodes. The signalling mechanism is out of the scope of this Thesis but it is assumed that 3 requested connections per update message is unaffordable from the point of view of the produced overhead. Moreover, if we consider the percentage of blocked connections, the number of established connections per update message will be lower than the number of requested connections per update message. Table 11 shows the number of established connections per update message for the *2-WSP_FA* algorithm when the traffic load is 10%, 15%, 20% and 25% of requested bandwidth. From Table 11 we observe that when traffic load increases the number of established connections per update message decreases because with more traffic load more blocked connections are produced. Moreover, when N increases (updates are more infrequent) the number of established connections per update message decreases too. This is due to the blocked connections produced by the inaccurate network state information utilized by the *WSP* algorithm.

Table 11: Number of established connections per update message for the *2-WSP_FA* algorithm.

N Traffic load	1	5	10	20
10%	3,01	15,08	30,00	59,19
15%	2,89	14,33	27,46	54,49
20%	2,46	12,18	23,00	46,73
25%	2,05	9,90	19,44	39,50

PART IV

CONCLUSIONS AND FUTURE WORK

This part concludes the Thesis summarizing the main goals achieved on it and proposing new open issues related to the ideas presented on this Thesis.

13. Summary and Conclusions

This Thesis proposes the Prediction-Based Routing (*PBR*) mechanism to tackle both the *RWA* problem in *WDM* networks and the *QoS* routing problem in *IP/MPLS* networks, aiming at solving the signalling overhead problem while reducing the effects of routing under inaccurate routing information The main characteristic of the *PBR* is to provide source nodes with the capability of taking routing decisions regardless of the global network state information obtained from the update messages. The novel idea introduced in the Prediction Based Routing allows the routes or lightpaths to be computed not according to the potentially inaccurate network state information but according to a prediction scheme. This prediction scheme is based on branch prediction concepts used in computer architecture. In this area the outcome of the branch instructions is not computed from the exact processor information but it is predicted using two-bit counters. The two-bit counters have 4 values: 0, 1, 2 and 3. In branch prediction the 0 and 1 values predict that the branch will be taken; and 2 and 3 predict that the branch will not be taken. Bringing this concept to a network scenario, the network state information is not obtained from the flooding of update message, but it is inferred from the behaviour of previous connection requests. The *PBR* mechanism takes into account the previous blocked connections produced in the same route or lightpath to train the two-bit counters. Now, the 0 and 1 values account for the availability of the route or lightpath; and 2 and 3 account for the unavailability. One important characteristic of the *PBR* is its simplicity compared with previous proposed mechanisms. One two-bit counter for route or lightpath (route and wavelength) suffices to implement the *PBR* mechanism.

Two immediate benefits may be inferred from the *PBR* mechanism. The former, the *PBR* removes the messages required to update the available network information located at the network state databases. The latter, the *PBR* reduces the connection blocking probability produced by the routing inaccuracy problem.

The *PBR* has been evaluated in different scenarios, Optical Networks, i.e. flat and hierarchical *WDM* Networks, Multilayer networks and *IP/MPLS* networks.

The Route and Wavelength Prediction (*RWP*) algorithm inferred from the PBR mechanism for optical networks has been compared with a usual *RWA* algorithm that extracts the network state information from update messages, the Shortest Path combined with Least Loaded (*SP-LL*). The results show that for affordable update frequencies the *RWP* algorithm outperforms the *SP-LL* algorithm. Only when the traffic load is too high for the network resources (wavelengths and fibres) the *RWP* degrades its performance. Moreover, it is shown the improvement in the *RWP* results when only 2 link disjoint routes are selected.

For hierarchical optical networks, this Thesis shows the benefit of using a *RWA* algorithm based on the *PBR* mechanism such as the Predictive Hierarchical Optical Routing (*PHOR*) algorithm. A hierarchical network is divided into Routing Areas (*RAs*) containing nodes with similar characteristics. In a hierarchical network the inaccuracy of network state information used by usual *RWA* algorithm is larger than in a flat network due to the needed of aggregating this information to be disseminated between the different *RAs*. The schemes of aggregation are used to reduce the signalling overhead produced by the flooding of this network state information. Two benefits are inferred of using the *PBR* mechanism in a hierarchical network. On one hand, it is not necessary to aggregate and flood the network state information; on the other hand the blocking of connection requests produced by the use of inaccurate information is reduced. Simulations in this area show that a *RWA* algorithm only based on prediction concepts is the best option for low traffic load. When traffic load is high, this Thesis proposes hybrid solutions, the Balanced Predictive Hierarchical Optical Routing (*BAPHOR*) algorithm. The proposal is a *RWA* algorithm based on both prediction and load balancing concepts. Results show that this is the best option with high traffic load. This hybrid scheme needs the update of network state information into the Routing Areas (*RAs*), but it is not necessary to aggregate and disseminate this information between different RAs.

The *PBR* mechanism has also been applied to the optical layer of a Multi-layer scenario, *IP* over *WDM*. In the *MTE* (Multi-layer Traffic Engineering) strategy every *IP* connection request is translated into one or more optical connection requests in the optical layer (from 1 to 16). For this reason a usual *RWA* algorithm which uses the network state information obtained from the update messages will be clse dependent on the update frequency. At affordable update frequencies these *RWA* algorithms degrades rapidly its performance. An algorithm based on the *PBR* mechanism has been proposed as *RWA* algorithm on the optical layer of a Multi-layer scenario. Simulations show the benefit of using the *PBR* mechanism in a Multi-layer scenario.

Finally, the *PBR* mechanism has been proposed for *IP/MPLS* networks. As in optical networks, the first proposal took into account some registering of the history of every route. But the evaluation of results showed that one two-bit counter for route suffices to implement the *PBR* mechanism. The different algorithms inferred from the *PBR* mechanism for *IP/MPLS* networks have been compared to the Widest Shortest Path (*WSP*) algorithm. This algorithm utilizes the network information coming from update message to compute the widest shortest available path. Results show the improvement of using the *PBR* mechanism in *IP/MPLS* networks, especially for low and medium traffic load. Moreover, it has been shown the importance of the selection of the alternate or candidate routes. The routing algorithms in general and the routing algorithms inferred from the *PBR* mechanism in particular, improve their performance when selecting among appropriate precomputed routes. An additional benefit is obtained when the *PBR* is used in *IP/MPLS* networks, the reduction of the signalling overhead. In optical networks, it is possible an out-of-band control network, but in *IP/MPLS* the signalling messages are delivered in the same data network. Using the data network to flood signalling information wastes network resources making congestion easier. When the *PBR* mechanism is utilized the signalling overhead produced by the flooded of update message is completely eliminated.

Summarizing, the *PBR* mechanism can be proposed as a good option to perform *RWA* and *QoS* routing in *WDM* and *IP/MPLS* networks respectively. The *PBR* mechanism utilizes the network state information obtained in previous connection requests and local information to select the lightpaths or routes; and then eliminating the update messages without affecting on the global network performance.

14. Future Work

Two futures lines of work might be developed from this Thesis. On one hand, the enhancement of the *PBR* mechanism and the development of new algorithms inferred from the *PBR* mechanism. The aspect of the *PBR* mechanism more suitable to be enhanced is the way that the two-bit counters are unblocked. In the mechanism presented in this Thesis, when no route or lightpath can be selected because all the two bit counters are larger than 1, the *PBR* mechanism assigns the first route with output link availability. If this route is selected and the connection is established, the corresponding two-bit counter will be decreased and unblocked. This form of unblocking the two bit counters is simple and it works, but it is possible to be refined. Concerning to the *RWA* or *QoS* routing algorithms, results in hierarchical networks show that hybrid solutions like the *BAPHOR* algorithm work very well. The *BAPHOR* algorithm is a hybrid solution between the pure predictive algorithm inferred from the *PBR* mechanism and a load balancing approach. The good results of the hybrid approach suggest that other hybrid algorithm can be designed for flat *WDM* networks, *IP/MPLS* networks and Multi-layer networks.

On the other hand, in this Thesis it is presented the *PBR* mechanism applied to optical networks (*WDM*), hierarchical optical (*WDM*) networks, in the optical layer of Multi-layer networks and in *IP/MPLS* networks. Open issues are to apply the *PBR* mechanism in the *IP* layer of a Multi-layer network. Even it would be suitable to design a complete Multi-layer strategy based on predictive concepts. In that case the routing algorithms in the *IP* and the optical layer would be inferred from the *PBR* mechanism. Moreover the Multi-layer strategy of communication between the two layers would be based too on a predictive approach. This communication between the two layers consists of deciding when the *IP* layer requests an optical connection.

Finally the *PBR* mechanism was designed to be applied to connection oriented networks, or circuit switched networks, but it can be modified to be applied to packet or burst switched networks. Especially interesting is the case of applying the *PBR* mechanism to an optical burst switched network because it can be considered such as a very dynamic circuit switched network.

REFERENCES

[1] A. S. Tanenbaum, “Computer Networks”, Upper Sadle River, NJ Pearson Education cop. 2003

[2] R. Ramaswami and K.N. Sirvajan, “Optical Networks: A Practical Perspective”, The Morgan Kaufmann Series in Networking, 2002.

[3] ITU-T G.804 Recommendation (1993), “ATM, Cell Mapping into Plesiochronous Digital Hierarchy (PDH)”.

[4] E.C. Rosen A. Viswanathan and R. Callon, “Multiprotocol Label Switching Architecture”, IETF RFC 3031, July 2001.

[5] ANSI T1.105-2001, Synchronous optical network (SONET): Basic description including multiplexing structure, rates and formats, 2001

[6] ITU-T G.702 Recommendation (1988), “Digital hierarchy bit rates”

[7] E. Mannie, “Generalized Multiprotocol Label Switching Architectures”, RFC 3945, October 2004.

[8] IEEE 802.3z (Gigabit Ethernet Standard), “Supplement to Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications: Media Access Control (MAC) Parameters, Physical Layer, Repeater and Management Parameters for 1000 Mb/s Operation”, IEEE Standards Department, Piscataway, NJ, 1998

[9] ANSI X3T1 Standard (1997), “Information Technology-Single-Byte Command Code Sets CONnection (SBCON) Architecture” (formerly ANSI X3.296)

[10] D. Awduche, J. Malcom, J.Agogbua, M. O’Dell and J. McManus, “Requirements for Traffic Engineering over MPLS”, IETF RFC 2702, September 1999.

[11] S. Shenker and J. Wroclawski, “Network Element Service Specification Template”, IETF RFC 2216, September 1997.

[12] J. Wrocclawski, “The Use of RSVP with IETF Integrated Services”, IETF RFC 2210, September 1997.

[13] S. Blake et al, “An Architecture for Differentiated Services”, IETF RFC 2475, December 1998.

[14] R. Braden et al, “Resource Reservation Protocol (RSVP)- Version 1, Functional Specifications”, IETF RFC 2205, September 1997.

[15] A. Kaheel, T. Khattab, A. Mohamed and H. Alnuweiri, “Quality-of-Service Mechanisms in IP-over-WDM Networks”, IEEE Communications Magazine, 4(12), December 2002.

[16] W.Fawaz, “A Novel Protection Scheme for Quality of Service Aware WDM Networks”, ICC 2005.

[17] W. Fawaz, “Service Level Agreement and Provisioning in Optical Networks”, IEEE Communications Magazine, 42(1), January 2004.

[18] B. Daheb, G.Pujolle, “Quality of Service Routing for Service Level Agreement Conformance in Optical Networks”, Globecom 2005.

[19] J. Moy “OSPF-Version 2”, IETF RFC 2328, April 1998.

[20] W. Stallings, “Computer Architecture Organization and Architecture, Principles of Structure and function”, Macmillan Publishing Company, 1993.

[21] E.Marín-Tordera, X.Masip-Bruin, S.Sànchez-López, J.Solé-Pareta, J.Domingo-Pascual, “A New Prediction-Based Routing and Wavelength Assignment Mechanism for Optical Transport Networks”, in Proc. of QofIS’04, Barcelona, September 2004.

[22] R. Ramaswami and K.N. Sirvajan, “Routing and Wavelength Assignment in All-Optical Networks”, IEEE JSAC vol.3 nº5 October 1995.

[23] Z. Zhang and A. Campora, “A Heuristic Wavelength Assignment Algorithm for multi-hop WDM Networks”, IEEE/ACM Transactions on Networking 3(3) June 1995.

[24] D. Banejee and B. Mukherjee, “A Practical Approach for Routing and Wavelength Assignment in large wavelength-routed Optical Networks”, IEEE JSAC 14(5) June 1996.

[25] R. Ramaswami and K.N. Sirvajan, “Design of logical topologies for wavelength-routed Optical Networks”, IEEE JSAC 14(5) June 1996.

[26] R. Dutta and G.N. Rouskas, “A Survey of Virtual Topology for wavelength-routed Optical Networks”, Optical Networks Magazine 1(1) January 2000.

[27] ITU-T Rec.68080/4.1304, “Architecture for the Automatically Switched Optical Network (ASON)”, November 2001.

[28] Harari H., Massayuki M., Miyahara H., “Performance of Alternate Routing Methods in All-Optical Switching Networks”, IEEE INFOCONM’97 1997.

[29] Chan K.M. and Yun T.S.P., “Analysis of Least Congested Path Routing Methods in WDM Lightwave Networks”, IEEE INFOCOM’94 1994.

[30] L.Li and A.K. Somani, “Dynamic Wavelength Routing Using Congestion and Neighborhood Information”, IEEE/ACM Transactions on Networking vol.7, nº5, October 1999.

[31] A. Todimala and B. Ramamurthy, “Congestion-based Algorithm for Online Routing in Optical WDM Networks”, Communication, Internet and Information Technology 2003.

[32] B. Mukherjee, “Optical Communications Networks”, McGraw-Hill, New York 1997.

[33] S. Subramanian and R.A. Barry, “Wavelength Assignment in Fixed Routing WDM Networks”, ICC’97 1997.

[34] G. Jeong and E. Ayanoglu, “Comparison of Wavelength Interchanging and Wavelength Selective Cross Connects in Multiwavelength All-Optical Networks”, IEEE INFOCOM’96 1996.

[35] E. Karasan and E. Ayanoglu, “Effects of Wavelength Routing and Selection Algorithms on Wavelength Conversion Gain in WDM Optical Networks”, IEEE/ACM Transactions on Networking vol.6 nº 2 April 1998.

[36] X.Zhang and C.Qiao, “Wavelength Assignment for Dynamic Traffic in Multi-fiber WDM Networks”, Proc. of the 7th International Conference on Computer Communication, 1998.

[37] A. Birman and Kershenbaum. “Routing and Wavelength Assignment Methods in Single-Hop All-Optical Networks with Blocking”, IEEE INFOCOM’95 1995.

[38] H.Zhang, J.Pue, B. Mukherjee, “A Review of Wavelength Assignment Approaches for Wavelength-Routed Optical WDM Networks”, Optical Networks Magazine, January 2000.

[39] A. Mokhthar and M. Azizoglu, “Adaptive Wavelength Routing in All-Optical Networks”, IEEE/ACM Transactions on Networking vol.6 nº 7 1998.

[40] M. Kovacevic and A. Acampora, “Benefits of Wavelength Translation in All-Optical Clear Channel Networks”, IEEE JSAC vol.14 nº5, June 1996.

[41] B. Ramamurthy and B. Mukherjee, “Wavelength Conversion in WDM Networking”, IEEE JSAC vol.16, nº7, September 1998.

[42] Li B. and Chu X., “Routing and Wavelength Assignment vs. Wavelength Converter Placement in All-Optical Networks”, IEEE Optical Communications, August 2003.

[43] Chu X. and Li B., “Dynamic Routing and Wavelength Assignment in the Presence of Wavelength Conversion for All-Optical Networks”, IEEE Transaction on Networking, vol.13 nº3, June 2005.

[44] X. Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, D.Colle, “Routing and Wavelength Assignment under Inaccurate Routing Information in Networks with Sparse and Limited Wavelength Conversion”, IEEE GLOBECOM 2003, 2003.

[45] B. Zhou, M. Bassiouni, G. Li, “Routing and Wavelength Assignment in Optical Networks Using Logical Link Representation and Efficient Bitwise Computation”, Photonic Network Communications vol.10 nº 3, 2005.

[46] D. Bisbal et al, “Dynamic Routing and Wavelength Assignment in Optical Networks by means of genetic Algorithms”, Photonic Network Communications, vol.7 nº 1, 2004.

[47] V.T. Le, S.H. Ngo, X. Jiang, S. Horiguchi, M.Gou, “A Genetic Algorithm for Dynamic Routing and Wavelength Assignment in WDM networks”, Proc. Inter. Symp. Parallel and Distributed Processing and Applications, 2004.

[48] R.M. Garlic, R.Barr, “Dynamic Wavelength Routing in WDM networks via Ant Colony Optimization”, in Ant Algorithms, Springer-Verlag, 2002.

[49] V.T. Le, S.H. Ngo, X. Jiang, S. Horiguchi, M.Gou, “Ant-Based Dynamic Routing and Wavelength Assignment in WDM Networks”, Proc. International Conference on Embedded and Ubiquitous Computing (EUC2004), 2004.

[50] V.T. Le, X.Jiang, S.H. Ngo, S. Horiguchi, “Dynamic RWA Based on the Combination of Mobile Agents Technique and Genetic Algorithms in WDM Networks with Sparse Wavelength Conversion”, IEICE Transactions on Information and Systems, September 2005.

[51] J.P.Jue, G.Xiao, “Analysis of Blocking Probability for Connection Management Schemes in Optical Networks”, in Proc. of IEEE GLOBECOM 2001, 2001.

[52] K.Lu, G.Xiao, I.Chlamtac, “Blocking Analysis of Dynamic Lightpaths Establishment in Wavelength-Routed Networks”, IEEE ICC 2002

[53] J.Zhou, X.Yuan, “A Study of Dynamic Routing and Wavelength Assignment with Imprecise Network State Information”, ICPP Workshop on Optical Networks, 2002.

[54] K.Lu, G.Xiao, I.Chlamtac, “Analysis of Blocking Probability for Distributed Lightpath establishment in WDM Optical Networks”, IEEE/ACM Transactions on networking, Vol.13, n.1, February 2005.

[55] S.Shen, G.Xiao, T.H.Cheng, “Evaluating the impact of the link-state update period on the blocking performance of wavelength-routed networks”, OFC 2004

[56] K.Lu, G.Xiao, J.P.Jue, T.Zhang, S.Yuan, I.Chlamtac, “Blocking Analysis of Multifiber Wavelength-Routed Networks”, IEEE GLOBECOM 2004.

[57] J.Zheng and H. Mouftah, “Distributed Lightpath Control Based on Destination Routing in Wavelength-Routed WDM Networks”, Optical Networks Magazine July/August 2002, vol.3 nº4.

[58] S. Darisala, A. Fumagalli, P. Kothandaraman, M.Tacca, L.Valcarenghi, M.Ali, D. Eli-Dit-Cosaque, “On the Convergence of the Link-State Advertisement Protocol in Survivable WDM Mesh Networks”,ONDM’03, 2003.

[59] K.Lu, J.P.Jue, G.Xiao, I. Chlamtac, T.Ozugur, “Intermediate-Node Initiated Reservation (IIR): A New Signalling Scheme for Wavelength-Routed Networks”, IEEE JSAC, vol.21, nº8, October 2003.

[60] X.Masip-Bruin, R.Muñoz, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, G. Junyent, “An Adaptive Routing Mechanism for reducing the Routing Inaccuracy Effects in an ASON”, ONDM’03, 2003

[61] Smith J.E., “A study of branch prediction strategies”, In Proc. of 8th International Symposium in Computer Architecture, Minneapolis 1981.

[62] E. Marín-Tordera, X. Masip Bruin, S. Sánchez-López, J. Solé Pareta and J. Domingo-Pascual, “The Prediction-Based Routing in Optical Transport Networks”, Computer Communications vol.29 issue 7, April 2006.

[63] H.Zang, J.P.Jue, L. Sahasrabuddhe, B. Mukherjee, “Dynamic Lightpaht Establishment in Wavelength-Routed WDM Networks”, IEEE Communications Magazine, September 2001.

[64] Deliverable D1.1 “Architectural vision of network evolution”, IST IP NOBEL Phase 2, “Next generation optical networks for broadband European leadership Phase 2”

[65] Sergio Sánchez López, "Interconnection of IP/MPLS Networks Through ATM and Optical Backbones using PNNI Protocols", Ph. D Thesis, October 2003.

[66] E. Marín-Tordera, X. Masip Bruin, S. Sánchez-López and J. Solé Pareta, "A hierarchical routing approach for optical transport networks", Computer Networks 50(2), 2006.

[67] E. Marín-Tordera, X. Masip Bruin, S. Sánchez-López and J. Solé Pareta, "Efficient Routing Algorithms for Hierarchical Optical Transport Networks", ECOC 2005, Glasgow.

[68] Xavier Masip Bruin, "Mechanisms to Reduce Routing Information Inaccuracy Effects: Application to MPLS and WDM Networks", Ph. D Thesis, October 2003.

[69] X.Masip-Bruin, S.Sànchez-López, J.Solé-Pareta, J.Domingo-Pascual, D.Colle, "Routing and Wavelength Assignment under Inaccurate Routing Information in Networks with Sparse and Limited Wavelength Conversion", in Proc. IEEE GLOBECOM 2003, San Francisco, USA, December 2003

[70] OIF2000.125.5, "User Network Interface (UNI) 1.0 Signalling Specification", June 2001.

[71] B. Puype, Q.Yan, D. Colle, S. De Maeesschalck, I. Lievens, M. Pickavet, and P. Demeester, "Multi-layer Traffic Engineering in Data-centric Optical Networks, Illustration of concepts and benefits", COST266/IST OPTIMIST Workshop, Budapest, Hungary 2003; pp. 211-226.

[72] G. Newsome, "ASON Characteristics" OIF contr.OIF2000.232 (2000)

[73] Bart Puype, Qiang Yan, Sophie De Maesschalck, Didier Colle, Kris Steenhaut, Mario Pickavet, Ann Nowé, Piet Demeester, "Optical cost metrics in Multi-layer Traffic Engineering for IP-over-Optical networks", ICTON 2004, Wrocław (2004), vol. 1; pp. 75-80

[74] Qiang Yan, Sophie De Maesschalck, Didier Colle, Bart Puype, Ilse Lievens, Mario Pickavet, Piet Demeester, "Influence of the observation window size on the performance of multi-layer traffic engineering", ITCOM, Orlando (2003), Proc. of SPIE Vol. 5247; pp. 203-214

[75] R. Guerin, A.Orda and D. Williams, "QoS Routing Mechanism and OSPF Extensions", in Proceedings of 2nd Global Internet Miniconference (joint with GLOBECOM'97) 1997.

[76] Z. Wang, J. Crowcroft, “Quality of Service Routing for supporting Multimedia Applications”, IEEE JSAC 14(7), September 1996.

[77] M. Kodialam, T.V. Lakshman, “Minimum Interference Routing with Applications to MPLS Traffic Engineering”, IEEE INFOCOM 2000.

[78] S. Suri, M. Waldvogel, P.R. Warkhede, “Profile-Based Routing: a new framework for MPLS Traffic Engineering”, QofIS 2001.

[79] Y.Yang, J.K. Muppala, S. T. Chanson, “Quality of Service Routing Algorithms for Bandwidth-Delay Constrained Applications” IEEE ICNP 2001.

[80] J.A. Khan, H.M. Alnuweiri, “A Fuzzy Constrained-Based Routing Algorithm for Traffic Engineering”, GLOBECOM’04, 2004.

[81] P.Van Mieghem, F.A.FKuipers, T.Korkmaz, M.Krunz, M.Curado, E.Monteiro, X.Masip-Bruin, J.Solé-Pareta, S.Sánchez-López, ‘Quality of Service Routing’Chapter 2 off boork "QUALITY OF FUTURE INTERNET SERVICES: COST 263 FINAL REPORT", Ed. Springer-Verlag, October 2003.

[82] R.A.Guerin, A.Orda, “QoS Routing in Networks with Inaccurate Information: Theory and Algorithms”, IEEE/ACM Transactions on Networking, vol.7 nº 3, June 1999.

[83] D.H. Lorenz, A. Orda, “QoS Routing in Networks with Uncertain Parameters”, IEEE/ACM Transactions on Networking, vol.6 nº6, December 1998.

[84] G.Apostolopoulos, R.Guerin, S.Kamat, S.K.Tripathi, “Improving QoS routing performance under inaccurate link state information”, Proc. ITC’16, 1999.

[85] S.Chen, K.Nahrstedt, “Distributed QoS routing with imprecise state information”, Proc.7th IEEE International Conference of Computer, Communications and Nettworks, 1998.

[86] X.Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, “A QoS Mechanism for Reducing the Routing Inaccuracy Effects”, Proceedings of the Second International Workshop on Quality of Service in Multiservice IP Networks, QoSIP 2003.

[87] X.Masip-Bruin, S.Sánchez-López, J.Solé-Pareta, J.Domingo-Pascual, “QoS Routing Algorithms under Inaccurate Routing Information for Bandwidth Constrained Applications”, Proc. IEEE ICC 2003.

[88] T.Korkmaz, M.Krunz, "Bandwidth-Delay Constrained Path Selection Under Inaccurate State Information", IEEE/ACM Transactions on Networking, Vol.11, nº3, June 2003.

[89] T. Anjali and C. Scoglio, “Traffic Routing in MPLS Networks Based on QoS Estimation and Forecast”, GLOBECOM’04, 2004.

[90] C. Scoglio, T.Anjali, J.de Oliveiro, I.Akyildiz and G. Uhl, “TEAM: A Traffic Engineering Automated Manager for DiffServ-based MPLS Networks”, IEEE Communications Magazine, vol.42, nº 10, 2004.

[91] Y. Yia, I.Nikolaidis and P. Gburzynski, “Multiple Path Routing in Networks with Inaccurate Network State Information”, IEEE ICC’01, 2001.

[92] Y. Yia, I.Nikolaidis and P. Gburzynski, “Scalable QoS Routing Using Alternative Paths”, International Journal of Communication System, vol.7, nº1, 2004.

[93] G. Rétvari, J.J. Bíró, T. Cinkler and T. Henk, “A Precomputation Scheme for Minimum Interference Routing: the Least-Critical-Path-First Algorithm”, INFOCOM 2005.

[94] P. Baran, “On Distributed Communications Networks”, IEEE Transactions on Communications (1964), 1-9.

[95] C. Busch, M. Herlihy and R. Wattenhofer, “Routing without Flow Control”, Proceeding of the thirteenth annual ACM Symposium on Parallel Algorithms and Architectures, 2001.

[96] T. Anjali, C. Scoglio, J. de Oliveira, L.C. Chen, I.F. Akyldiz, J.A. Smith, G. Uhl and A. Sciuto, “A New Selection Algorithm for MPLS Networks Based on Available Bandwidth Estimation”, QofIS 2002.

[97] S. Nelakuditi, Z. Zhang, R. P. Tsang and D. Du, “Adaptive Proportional Routing: A Localized QoS Routing Approach”, IEEE/ACM Transactions on Networking (ToN) vol.10, i.6, December 2002.

[98] S. Nelakuditi, Z. Zhang, and D. Du, “On Selection of Candidate Paths for Proportional Routing”, Computer Networks: The International Journal of Computer and Telecommunications Networking, v.44, n.1, January 2004.

[99] J. Foag and T. Wild, “Traffic Prediction for Speculative Network Processors”, HPCS’04, 18th International Symposium on High Performance Computing Systems and Applications, 2004.

[100]Y. G. Kim, A. Shiravi and P. S. Min, “Prediction-Based Routing through Least Cost Delay Constraint”, Proceeding of the IEEE International Parallel and Distributed Processing Symposium 2004 (IPDPS’04).

[101] E.Marín-Tordera, X.Masip-Bruin, S.Sánchez-López, “Prediction-Based Routing in IP/MPLS Networks“, Infocom 2005 Student Workshop, 2005.

[102] E. .Marín-Tordera, X.Masip Bruin, S.Sánchez-López, J. Domingo-Pascual, “The Prediction Approach in QoS Routing”, ICC 2006.

[103] Mitra and Seery, “Comparative Evaluations of Randomized and Dynamic Routing Strategies for Circuit-Switched Networks”, IEEE Trans. on Communications, pp. 102-116, 1991.

MAIN PUBLICATIONS

Journals

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López, Josep Solé Pareta and Jordi Domingo-Pascual, “The Prediction-Based Routing in Optical Transport Networks”, Computer Communications vol.29, issue 7, April 2006.

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López and Josep Solé Pareta, “A hierarchical routing approach for optical transport networks”, Computer Networks 50(2), 2006.

Conferences

2007:

- Eva Marín-Tordera, Xavier Masip Bruin, Sergio Sánchez López, Josep Solé Pareta, Guido Maier, Walter Erangoli, Stefano Santoni and Marco Quagliotti, ‘Applying Prediction Concepts to Routing on Semi-Transparent Optical Transport Networks’, ICTON 2007, Rome (Italy) July 2007.

2006:

- Eva Marín-Tordera, Xavier Masip Bruin, Sergio Sánchez López, Jordi Domingo Pascual and Ariel Orda, “The Prediction Approach in QoS Routing”, ICC 2006, Istambul, June 2006.

2005:

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López and Josep Solé Pareta, “Efficient Routing Algorithms for Hierarchical Optical Transport Networks”, ECOC 2005, Glasgow.
- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López, “Prediction-Based Routing in IP/MPLS Networks“, INFOCOM 2005, Student Workshop, Miami March 2005.

2004:

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López, Josep Solé Pareta, Jordi Domingo Pascual, “A New Prediction-Based Routing and Wavelength Assignment Mechanism for Optical Transport Networks”, in Proc. of QofIS’04, Barcelona, September 2004.

OTHER PUBLICATIONS

- Marcelo Yannuzzi, Xavier Masip Bruin, Sergio Sánchez López, Eva Marín Tordera, Josep Solé Pareta and Jordi Domingo Pascual, "Interdomain RWA based on stochastic estimation methods and adaptive filtering for optical networks", GLOBECOM 2006.

- Xavier Masip Bruin, Sergio Sanchez López, Eva Marín Tordera, Josep Sole Pareta. "The PBR Approach: Analysis, Performance and Perspective", ICTON 2006 .

- Sergio Sanchez Lopez, Xavier Masip Bruin, Eva Marin Tordera, Josep Sole Pareta, Jordi Domingo Pascual "A Hierarchical Routing Approach for GMPLS based Control Plane for ASON", ICC 2005.

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López and Jordi Domingo Pascual, "Prediction-Based Routing in IP-MPLS Networks", IV Workshop in G/MPLS Networks 2005.

- Eva Marín Tordera, Xavier Masip Bruin, Sergio Sánchez López, Josep Solé Pareta, "Mecanismo de encaminamiento y asignación de longitud de onda para Redes de Transporte Óptico", Telecom I+D 2004.

- Xavier Masip Bruin, Sergio Sánchez López, Josep Solé Pareta, Jordi Domingo Pascual, Eva Marín Tordera, "Hierarchical Routing with QoS Constraints in Optical Transport Networks", Networking 2004.