# OBS network model for OMNeT++: A performance evaluation

Felix Espina
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
felix.espina@unavarra.es

Javier Armendariz
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
javier.armendariz@unavarra.es

Naiara García
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
naiara.garcia@unavarra.es

Daniel Morató
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
daniel.morato@unavarra.es

Mikel Izal
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
mikel.izal@unavarra.es

Eduardo Magaña
Public University of Navarre
Campus de Arrosadía s/n
E-31006 Pamplona, Spain
eduardo.magana@unavarra.es

## ABSTRACT

Optical Burst Switching (OBS) is an optical switching technology capable of supporting large demands for bandwidth in optical backbones with Wavelength Division Multiplexing (WDM). This paper presents an OBS simulation model for the discrete event simulator OMNeT++. The performance of this model is compared with the performance of the well-known INET simulation model for IP networks. Both models show similar performance results. The OBS model is faster but uses more dynamic memory.

## Categories and Subject Descriptors

C.2.5 [**Computer-Communication Networks**]: Local and Wide-Area Networks—*High-speed*; I.6.5 [**Simulation and Modeling**]: Model Development; I.6.8 [**Simulation and Modeling**]: Types of Simulation—*Discrete event*; D.3.2 [**Programming Languages**]: Language Classifications—*Specialized application languages*

## General Terms

Design, Performance

## Keywords

Optical Burst Switching, OBS, OMNeT++

## 1. INTRODUCTION

### 1.1 Introduction to OBS

Optical Burst Switching (OBS) [13] has received considerable research attention as a promising solution for all-optical transmission of data. It allows the creation of high-speed all-optical networks with present technology, compared to Optical Packet Switching (OPS) that still suffers serious technological limitations [23] and Optical Circuit Switching (OCS) that has limited traffic adaptation and poor bandwidth utilization for bursty traffic [24]. In fact, the number of OBS testbeds and implementations has grown considerably in recent years [2][27][9][15][6][16].

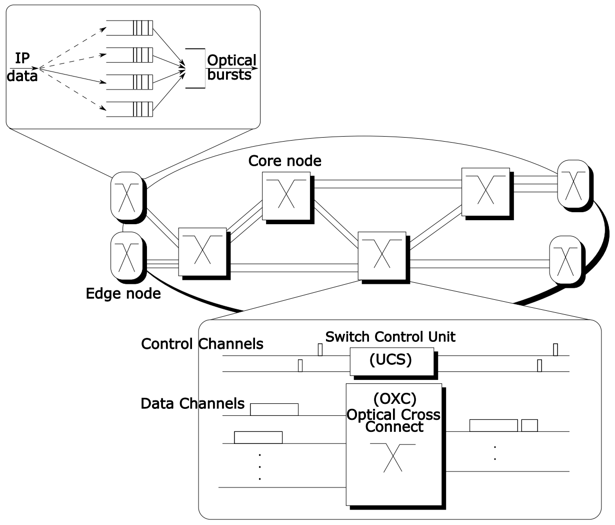

Figure 1: OBS network architecture

In an OBS network (Fig. 1) the packets from legacy networks are buffered at the ingress nodes and aggregated into bursts based on Forward Equivalence Classes (FECs) in what is often called a *burstifier*. Therefore, an OBS simulation model should allow the interaction between a packet data network (e.g. and Ethernet LAN) and an OBS cloud, for example for the transport of end-to-end IP packets. The model presented in this paper allows this interaction.

Although one FEC is normally created for each destination egress node, priority classes could be implemented with different FEC per class or using class differentiation inside the burst [14][21]. These bursts are optically switched by

*Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8708*
*http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8708*

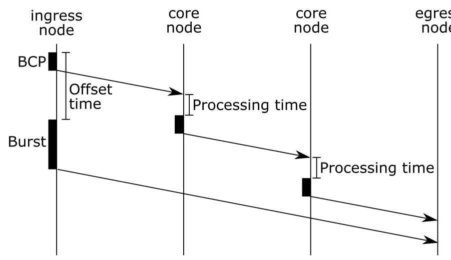

**Figure 2: Offset time and processing of BCP in core nodes**

the core nodes in the network and disassembled at the egress node in order to be relayed to the destination.

The prevalent burstifier types in the literature are timer-based, size-based or a mixture of both timer- and size- based [4]. In a timer-based burstifier a timer is started on the arrival of a packet to an empty burst formation queue. When the timer expires, the burst is scheduled for transmission on the output port. In a size-based burstifier the burst is sent when the planned minimum size is reached. Mixed-timer-size burstifiers complete the burst when at least one of the conditions is true: timer expiration or minimum size reached. Other less popular proposals in the literature use predictive or adaptive techniques in order to reduce the delay and losses [10, 7].

A Burst Control Packet (BCP) is created and sent by the ingress node an offset time before the burst is sent (see Fig. 2). This optical packet is electronically switched and processed at every backbone node. It contains information that depends on the signaling solution used, for example the burst arrival time, burst size, destination, etc. [17]. With this information, the backbone nodes decide the appropriate forwarding path for the optical burst from the ingress node to the egress node.

A WDM (Wavelength Division Multiplexing) solution is used at the fiber links (Coarse WDM, Dense WDM, ...). A wavelength is reserved for the transmission of the BCPs at each link while the remaining ones are used for data transport.

Generally OBS uses one-way signaling schemes initiated by the source. Bursts are sent into the OBS network without waiting for response about the success of the reservation of a full path to the destination. Thus, the bursts may compete for the same resources when they reach a switching node. This is the main cause of burst loss in OBS. These losses occur when the number of colliding attempts of burst reservations at a switching node output port exceeds the available number of wavelengths. It also happens when the BCP and its associated burst are too near in time. The core node does not have time to program the switch and has to discard the burst. Either way, this implies the loss of the burst.

These losses can be minimized using optical buffers on the backbone nodes or using techniques such as deflection routing. The first one can only be accomplished nowadays using fiber delay lines, whose performance differs significantly from the one obtained by traditional buffering. The latter needs to be studied in greater depth, taking into account the impact that the deflection could have in the whole network, to ensure that the overall network performance is improved and not only the performance of a node.

## 1.2 Network Simulators

This paper presents a model for the OMNeT++[1] network simulator with which to study this new technology. Currently, there are proposals for the different parts of an OBS network (timer or size based burstifiers, signaling techniques) but very few testbeds. Simulation is always used at this stage for network architecture design and parametrization.

There are many network simulators [20] that have different characteristics and purpose. For this work OMNeT++ version 3.3 [19] was chosen. Unlike others, the main function of OMNeT++ is not to be a network simulator, but to be a generic discrete event simulator framework with which to create simulators for different scenarios, from the operation of a hard disk to the behavior of an Ethernet network. This gives great versatility and ability to be exploited in various fields. OMNeT++ is open source and has a *Academic Public License* which makes it free for non-commercial use. It is available for all common platforms, including Windows, Mac OSX and Linux. All source code is in C++ and can be compiled with the compiler gcc or Microsoft Visual C++.

The other two network simulators most widely used in research are: OPNET[2] and NS-2[3].

OPNET has a very expensive annual license, even for scientific research. Paying a license gives access to the source of the models, but not to the source of the simulator's kernel. A significant difference is that OPNET models are always of fixed topology, while in OMNeT++ it is easy to have parametrized topologies. In OPNET the usual and preferred way to define the network topology is with the graphical editor, which keeps the network in a proprietary binary format that makes it difficult to generate topologies programmatically (you must use a specific C API). In contrast, in OMNeT++, topologies are stored in plain text files that are easy to manipulate. The main advantage of OPNET above OMNeT++ is its large library of protocol models (including one for OBS), while its closed nature makes programming and troubleshooting difficult.

NS-2 network simulator is the most widely used in academic research, but has not the separation between kernel and simulation models that OMNeT++ has. NS-2 distribution contains the models within the simulator infrastructure as one inseparable entity. NS-2 goal is to create a network simulator, while OMNeT++ goal is to provide a simulation platform. NS-2 lacks many tools and infrastructure components that OMNeT++ has: support for hierarchical models, graphical user interface (GUI) simulation environment, separation between models and experiments, graphical tools for analysis, some features like multiple simulation random number generator (RNG) streams, etc. NS-2 has focused on developing simulation models instead of a simulation infrastructure. Furthermore, while NS-2 is open source and multiplatform, in Windows it loses some functionality and you must compile and use it through Cygwin[4], a Linux environment for Windows.

Until now, although some OMNeT++ simulation models for OBS [1] have been proposed, none has been made pub-

[1]http://www.omnetpp.org
[2]http://www.opnet.com/
[3]http://nsnam.isi.edu/nsnam/
[4]http://www.cygwin.com/

*Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8708*
*http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8708*

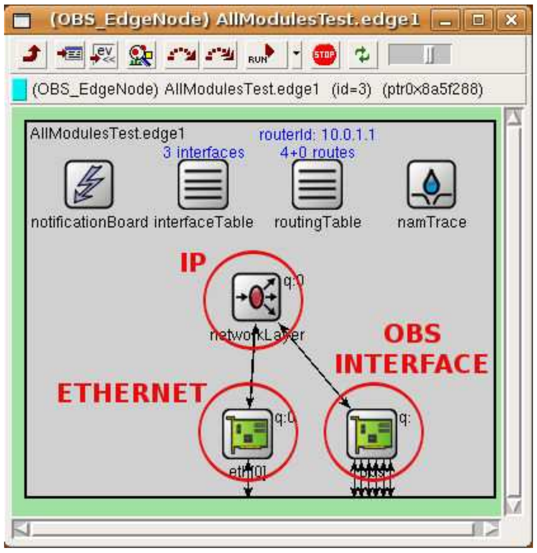

Figure 3: Edge Node

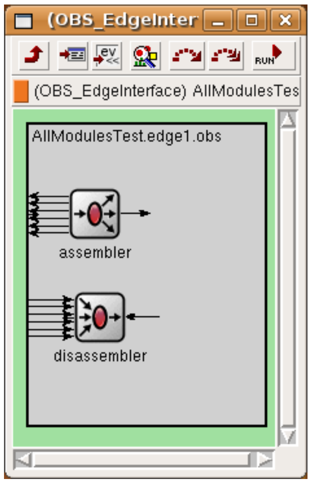

Figure 4: OBS interface of Edge Node

lic. Instead, a public OBS model exist for other simulators such as NS-2 [11]. This is the reason to develop the OMNeT++ modules for OBS presented in this paper (accessible via web[5]).

OMNeT++ provides the machinery and the basic tools to write those components and those simulations, instead of providing simulation components for computer networks, queuing networks and other domains. It is a framework rather than a simulation program. For each application specific area, specific models have been developed, such as the INET Framework for IP network simulation. The development of those models is completely independent from OMNeT++, including its publication cycle. Many models are developed by teams outside OMNeT++, although it is published in OMNeT++ website.

The rest of the paper is structured as follows. Section 2 describe the features of the developed OBS model for OMNeT++. Section 3 presents the OBS and Ethernet scenarios under analysis and the experimental setup for the performance comparison. Section 4 shows the results and discussion for an OBS scenario with different number of data wavelengths for fiber link but the same aggregated link capacity. In section 5 the OBS scenario and Ethernet-INET scenario are compared. Finally section 6 concludes the paper.

## 2. OBS NETWORK MODEL

The model assumes that there are two types of nodes: edge nodes and core nodes. The first one, the edge node, is the ingress and egress node to the OBS core network, using *burstifiers* to create bursts. Edge nodes are assumed not to transit burst traffic. The second one, the core node, is only responsible of traffic in transit, without the ability to introduce (or remove) traffic to the core network. This architecture is different from the one presented in [1], based on the theoretical proposal of [8], where all the nodes have the capacity to introduce (or remove) traffic to the OBS network and to have traffic in transit.

[5]https://www.tlm.unavarra.es/investigacion/proyectos/strrong/soft/

The model simulates correctly the basic operations of OBS. Validation of the implementation required the inspection of the low-level operation behaviour (bursts creation, bursts sent at the right time,...). Direct inspection validation was complemented by unit and system tests that are included with the software itself ("Test" folder).

In the next subsections, implemented OMNeT++ edge node and core node modules are described, highlighting their configuration capabilities and modularity. A preliminary version of this simulator without any performance evaluation was presented at JITEL2009 [5].

### 2.1 Edge Node

The edge node will be modelled as a router with an OBS interface (see Fig.3). The implementation is based on the OMNeT++/INET basic router module, named **Router**.

The parameters of interest are configured independently for each edge node. The network has to maintain some consistency, for example, it is not possible to connect a 5 wavelengths output port from an edge node to a just 3 wavelengths input port in a core node. Each wavelength used in an optical fiber was modelled as a separate OMNeT++ link, so OMNeT++ draws them separately. This is shown in figures 3 and 4, where there is only one optical fiber with 5 wavelengths (4 for data and one for signaling).

The edge node (Fig. 4) acts as an ingress node when it introduces traffic in the OBS network and as an egress node when it removes traffic from the OBS network.

When serving as an ingress node, the ***assembler*** module (Fig. 5) is responsible for assembling the incoming traffic into bursts and scheduling the transmission of the bursts into the output channel.

When serving as an egress node, the ***disassembler*** module performs the inverse operation, breaking down the incoming bursts into packets and forwarding them.

In OBS, incoming traffic is aggregated at the edge node into bursts depending on the optical destination. This aggregation takes place in burstifiers. There should exist at least one burstifier per destination (or egress node). There could be more than one burstifier per destination in order to differentiate traffic (and the generated bursts) depending on the source IP address or destination, the source or

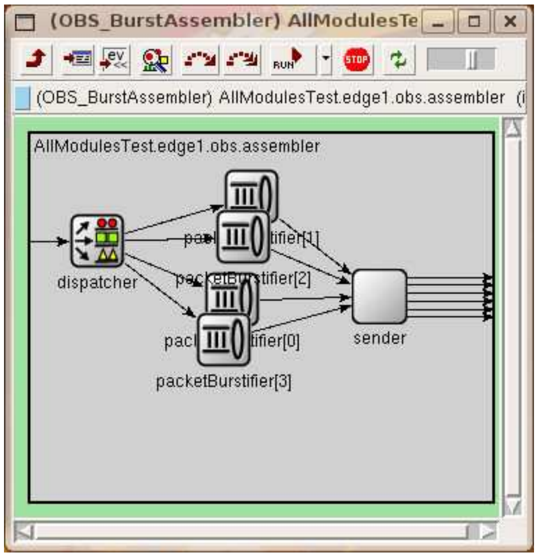
(OBS_BurstAssembler) AllModulesTe
(OBS_BurstAssembler) AllModulesTest.edge1.obs.assembler
AllModulesTest.edge1.obs.assembler
dispatcher
packetBurstifier[1]
packetBurstifier[2]
packetBurstifier[0]
packetBurstifier[3]
sender

**Figure 5: Assembler of Edge Node**

destination port and others characteristics of the traffic.

Other models [1] do not simulate the formation of bursts, since they only address the behavior of the OBS backbone. This approach does not allow the use of the OBS model in simulations that connect the OBS network with other data networks. The proposed model allows this. Moreover, the generated burst traffic can be studied depending on the incoming traffic and the aggregation schemes used in burstifiers. The current implementation of edge node supports the most common schemes: timer, size [4] and packet number thresholds [25]. And, of course, the mixture of these schemes. Adding a new scheme involves changing only one OMNeT++ simple module: the ***burstifier***.

Optical forwarding is done using a *label optical switching* type schema, similar to LOBS [12]. Each burst has a label. This label, together with the input port and wavelength, is used in the core nodes as forwarding parameters. The label may change at each hop. The burstifier at the edge node that generates the burst puts the initial label. The ***dispatcher*** of the OBS interface, using its parameters, decides in which of the burstifiers to store the incoming traffic. The label given to the generated burst is the label configured in the simulation for that burstifier.

Once the burst is generated, it is delivered to the OBS link level (***sender*** module). This link level has been implemented as a queue in which to store the bursts until their transmission. The size of this queue, in bits and number of bursts, is configurable for each simulation and edge node. When a burst cannot fit in the queue, it is discarded.

The OBS scheduler currently used is the most popular and basic scheme proposed, called **Horizon** or **LAUC** [18]. In this scheme, on a burst arrival the closest time when any of the wavelengths is free is calculated. The burst transmission is planned for that moment and that wavelength and the horizon of the wavelength are updated. A core node needs to process the control information of the BCP before the burst arrives, so the BCP associated with the burst is sent some offset time before the burst. The temporal offset between the BCP and the burst has a maximum and minimum limits. Initially, the BCP is planned to be sent with the maximum offset. If the control channel suffers congestion and the BCP can not leave the edge node with the minimum offset separation, the BCP and the burst are dropped.

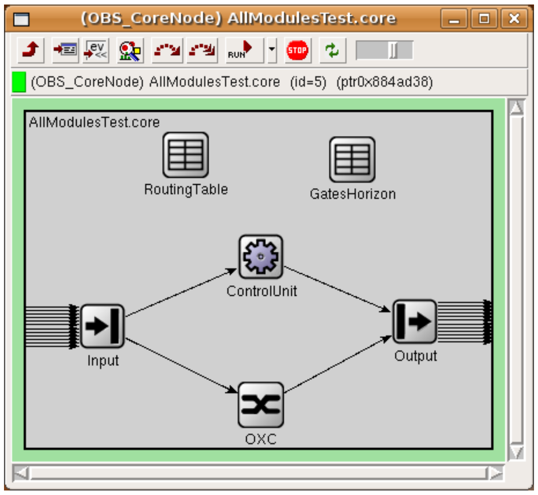
(OBS_CoreNode) AllModulesTest.core
(OBS_CoreNode) AllModulesTest.core (id=5) (ptr0x884ad38)
AllModulesTest.core
RoutingTable
GatesHorizon
ControlUnit
Input
Output
OXC

**Figure 6: Core Node**

## 2.2 Core Node

The core nodes are responsible of the BCPs processing, the bursts switching from an input fiber to another output one without electro-optical conversion, and the mechanism of contention resolution between bursts.

Figure 6 shows the internal structure of the core node. It is based on making forwarding of the bursts from a fiber to another through a dynamically configurable switching matrix. This switching matrix follows the orders of the control unit. The control unit makes its decisions using the *forwarding table*, which indicates the output port, wavelength and label according to the input port, wavelength and label of the burst. Currently, this forwarding table is generated from a file at initialization, and remains unchanged during the simulation. The implementation of the core node could be changed to include a dynamic forwarding table that changes its state, e.g. through a centralized routing scheme.

OBS signaling is typically made out of band, transmitting the BCP at an unique wavelength different from the wavelengths for bursts. Different signaling schemes have been proposed [17][26], but the most popular distributed signaling protocols in OBS are Just-In-Time (JIT) [22] and Just-Enough-Time (JET) [3]. Both are one-way signaling schemes initiated by the source, ie, bursts are sent to the OBS network without waiting for confirmation of the success or failure of the attempt to reserve a path to the destination. They are based on the same principle, but differ in the duration of the reservations. JIT uses immediate reservation since the BCP reaches the core node, while JET delays channel reservation to the estimated arrival of the burst. In JET the BCP must indicate when the burst is expected to arrive, so the signaling information required is different and the type of BCP used in JET and JIT is different.

The reservation delay makes JET more efficient than JIT, obtaining lower blocking ratios and smaller end-to-end delays [8]. The current implementation of this OBS simulation model uses a JET scheme, although it could easily change to a JIT scheme. It would only be necessary to change a

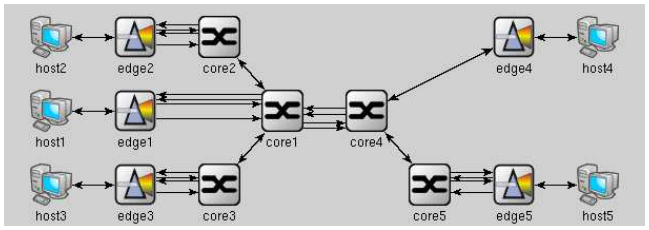

**Figure 7: OBS network scenario**

small part of the core node, the simple module ***ControlUnit***. There is no need to change the BCP message because the current message already has all the necessary fields for JIT signaling.

Each input port, wavelength and label has associated in the ***ControlUnit*** which output port, wavelengths and label can use or are *valid*. Upon a BCP arrival, the *valid* wavelength that has the horizon closer to and smaller than the estimated arrival time of the burst is selected. The ***Optical Cross-Connect (OXC)*** is scheduled to switch the input wavelength with the selected output wavelength at the arrival instant and to undo once the burst crosses the switching matrix. In the present model, a **guard time** between the estimated arrival time and the effective switching and unswitching times can be configured. Currently, the scheme assumes that there is always a wavelength converter available for this switching between wavelengths. If there is no free wavelength at the arrival moment, the BCP is discarded and when the burst arrives, it is lost.

## 3. NETWORK SCENARIO AND METHODOLOGY

Global performance compared with a similar simulation in NS2 (or other simulation framework) will not only depend on the quality of the presented OBS implementation and NS2 OBS implementation, but also on the different performance from NS2 to OMNeT++. The intention of this paper is to evaluate only the OBS implementation, not to evaluate the OMNeT++ simulation framework with other simulators frameworks. The performance of the developed OBS model is evaluated against a similar module for OMNeT++ so that both scenarios share a common ground (OMNeT++) and the difference is due to the code of the different technologies and implementations. In this case, the performance of the developed OBS model is evaluated against the performance of the well known INET model.

Two analogue network scenarios have been used, one for the OBS simulations (Fig. 7) and another one for the INET simulations (Fig. 8). In the INET scenario the switching technology that substitutes OBS is Ethernet.

Three performance parameters were measured:

- Duration of the simulation
- Number of events of the simulation
- Maximum amount of memory used by the simulation

These performance parameters cannot decided if one model is better than the other model, because they are different technologies and the exact numbers depend on their complexity. They show whether the OBS model has a serious penalty against other well-known OMNeT++ switching

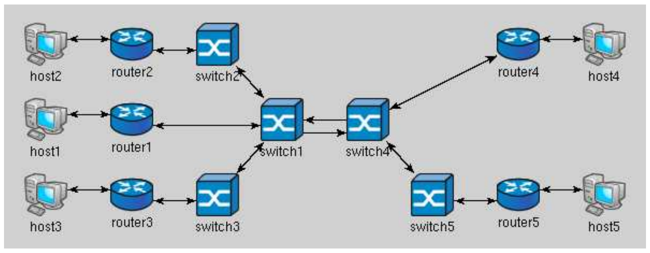

**Figure 8: INET scenario for comparison**

modules. By comparing CPU and memory usage, its scalability can be compared to existing models. That is, they show the “quality” of the new OBS model.

The ingress node to the OBS cloud is an IP router with an OBS interface. Inside the OBS interface, the burst formation mechanism is implemented. The timer-based burstifier is the most frequently used in the literature as it can easily guarantee a maximum delay for the traffic. So, in the OBS simulations timer-based burstifiers for the edge nodes have been used.

In both simulation scenarios, every host generates UDP traffic to all of the other hosts. This traffic has Poisson distribution arrivals and fixed packet lengths (maximum UDP packet length without IP fragmentation in an Ethernet network). The Poisson distribution parameters are chosen such that they create a preconfigured load at the central link (switch1-switch4 in Fig. 8 and core1-core4 in Fig. 7).

In the Ethernet scenario, the links are 10Gbps Ethernet links, while in the OBS scenario two different approaches have been taken:

- Only one data wavelength (10Gbps of capacity) per link.
- Ten data wavelengths (1Gbps of capacity) per link.

For the OBS simulations, the new modules presented in this paper were used. For the Ethernet simulations the INET network model of OMNeT++, modules *Router* and *EtherSwitch*, was used. For both scenarios the simulations run for 1 minute (*simulated time*). The stationary state is reached in less than 1 simulated second.

All the simulations were made using the same machine, an Intel Core 2 Duo E6570 (@2.66GHz) with 3GiB of RAM and Ubuntu 8.04.

## 4. THE EFFECT OF NUMBER OF WAVELENGTHS ON THE OBS LINKS

In OBS the same link capacity can be obtained using only one wavelength of that bitrate or a finite number of wavelengths of a lesser bitrate. Although they use the same technology, maybe it is more expensive (in events, time, etc.) to simulate one than the other.

Figure 9 presents the numbers of simulation events processed in the different OBS simulations (for different timer $T_{out}$ and number of wavelengths at the optical links) with different load of the central link between **core1** and **core4** versus the simulated time.

The number of events increases linearly with the load of the central link, because the number of packets generated by the hosts increases linearly. As the timer increases the

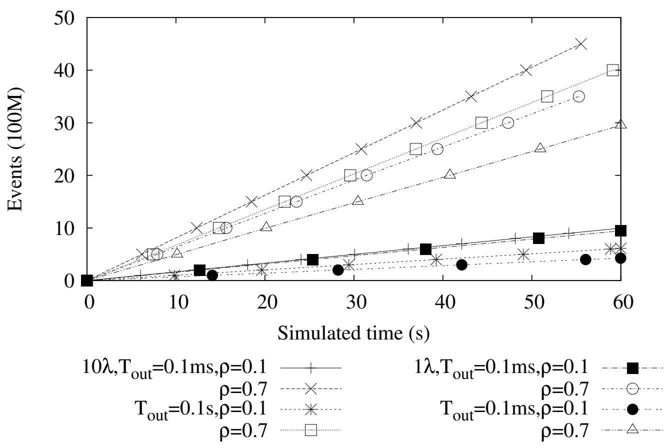

**Figure 9: Events of OBS scenario with 1 and 10 wavelengths, different timer and load in the central link vs simulated time**

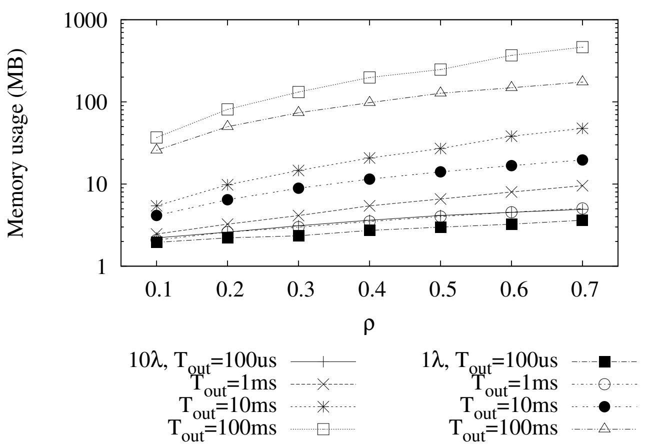

**Figure 11: Memory usage of OBS scenario with 1 and 10 wavelengths and different timer vs load in the central link**

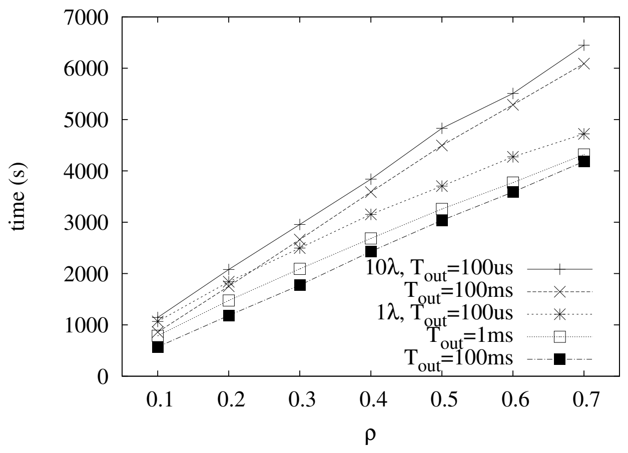

**Figure 10: Processing time of OBS scenario with 1 and 10 wavelength and different timer vs load in the central link**

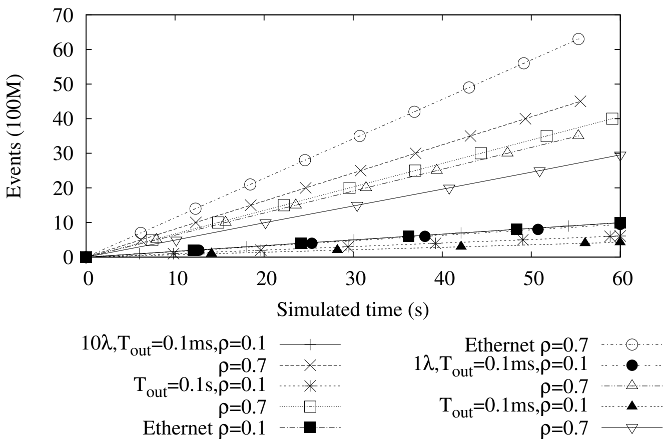

**Figure 12: Events of Ethernet scenario and OBS scenario with 1 and 10 wavelengths, different timer and load in the central link vs simulated time**

number of bursts gets reduced (more packets per burst), therefore the number of events gets smaller. It can be seen that the number of events grows with the number of wavelengths.

Figure 10 presents the processing time for the different OBS simulations (timer $T_{out}$ and the number of wavelengths of the optical links) versus the load of the central link between **core1** and **core4**.

As the figure shows the processing time grows linearly with the load. The simulation time also grows with the number of wavelengths. More events usually implies more time and more memory. The latter can be seen in figure 11 where the memory usage is plotted. But, the simulation duration grows when $T_{out}$ decreases.

Figure 11 presents the maximum memory allocation in the different OBS simulations (for different timer $T_{out}$ and number of wavelengths at the optical links) versus the load of the central link between **core1** and **core4**. The memory was measured recording every second the *pmap* command output. This command displays the process memory map. The memory grows with the load, because more load implies more scheduled events that implies the need for more memory. As the timer increases the number of packets inside each burst and the time they spend inside it grows, so more memory is needed. The memory also grows with the number of wavelengths.

Therefore, an OBS simulation with 10 wavelengths at 1Gbps costs more in events, simulation duration and memory used than a simulation with one wavelength at 10Gbps and the same load $\rho$ and timer $T_{out}$.

## 5. COMPARISON WITH INET

In this section the performance of the OBS simulator will be compared with the performance of the well know INET models for IP networks for the OMNeT++ framework.

Figure 12 presents the number of events processed in the Ethernet simulations and some OBS simulations (timer $T_{out}$ and the number of wavelengths of the optical links) with different load of the central link versus simulated time.

It is clear that INET models have at least the same num-

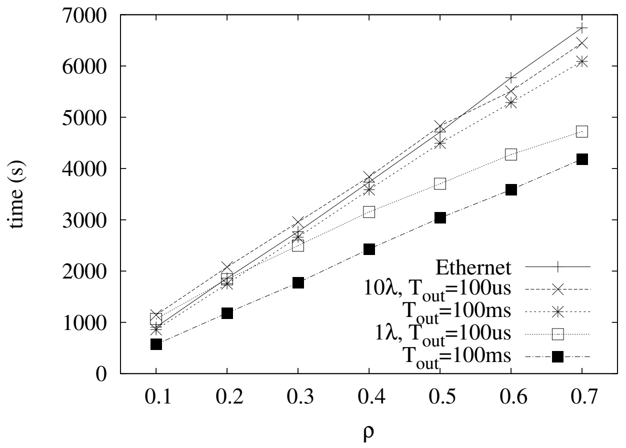

**Figure 13: Processing time of Ethernet scenario and OBS scenario vs load in the central link**

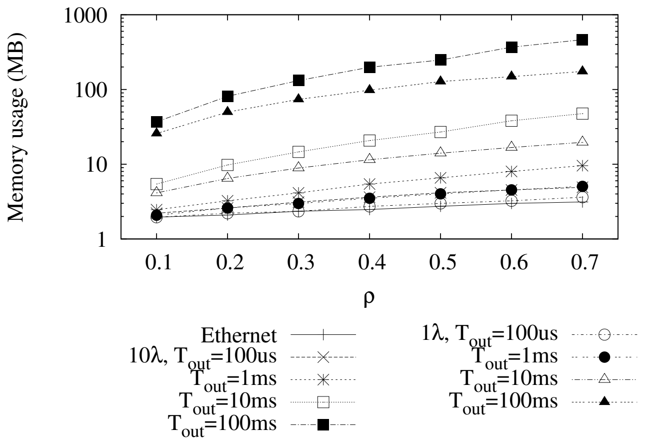

**Figure 14: Memory usage of Ethernet scenario and OBS scenario with 1 and 10 wavelengths and different timer vs load in the central link**

ber of events as the best equivalent OBS simulation, the simulation with the largest number of wavelengths and the smallest $T_{out}$. For the best scenario the number of bursts in the OBS core is similar to the number of IP packets in the Ethernet core, and so the number of events in both scenarios is similar. When the number of wavelengths decreases, the timer $T_{out}$ increases or the load $\rho$ increases, more packets are inside each burst and so less forwarding work has to be done inside the core network on the OBS model. For low load this difference is not as significative as for medium and high load.

Figure 13 presents the processing time of the Ethernet simulations and some different OBS simulations (timer $T_{out}$ and the number of wavelengths of the optical links) versus the load of the central link between **core1** and **core4**.

For small $\rho$ the INET model is as fast as the OBS model with one wavelength for each link and faster than the OBS model with 10 wavelengths for each link. For moderate to high $\rho$ the Ethernet model is always worse than the OBS model, because the number of events to manage is always greater. This can be seen in figure 12. For low load this difference is not as significative as for medium and high load.

Figure 14 presents the maximum memory allocation for the Ethernet simulations and some OBS simulations (timer $T_{out}$ and the number of wavelengths of the optical links) versus the load of the central link. The INET model uses always less memory than the OBS model. In OBS, packets travel in groups inside bursts, so they spend more time inside simulator and therefore increase the memory usage.

## 6. CONCLUSIONS

In this paper a new OBS model for OMNeT++ discrete events framework was introduced. This model includes the implementation of both edge node and core node. It was developed and implemented taking into account modularity to allow the addition of future proposals of this technology.

Although it is the first development, the model simulates correctly the basic operations of OBS.

The performance of the OBS model was compared with the well know INET model and it has similar performance in number of events and simulation duration, although OBS model uses need more memory.

## 7. ACKNOWLEDGMENTS

This work was supported by MEC (project STRRONG TEC2007-62192/TCM). The authors wish to thank the thematic network IPoTN (TEC2008-02552-E/TEC).

## 8. REFERENCES

[1] A. L. Barradas and M. C. R. Medeiros. An OMNeT++ model for the evaluation of OBS routing strategies. In *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, pages 1–7, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[2] A. Campi, W. Cerroni, F. Callegati, G. Zervas, R. Nejabati, and D. Simeonidou. SIP Based OBS networks for Grid Computing. In *LNCS 4534, Proceedings of ONDM*, Athens, Greece, May 2007.

[3] Y. Chen, C. Qiao, and X. Yu. Optical burst switching: a new area in optical networking research. *Network, IEEE*, 18(3):16–23, May-June 2004.

[4] J. Choi, J. Choi, and M. Kang. Dimensioning Burst Assembly Process in Optical Burst Switching Networks. *IEICE Transactions on Communications*, E88-B(10):3855–3863, October 2005.

[5] F. Espina, J. Armendariz, M. Izal, D. Morató, and E. Magaña. Arquitectura y diseño de un modelo de red OBS para simulación. In *Jornadas de Ingeniería Telemática (JITEL)*, September 2009.

[6] Z. Gao, Y. Qiao, Y. Ji, and T. Saito. The Optical Burst Switching Ring Network Using AOTF to Drop Data Burst. *Journal of Optical Communications*, 26(6):255–259, 2005.

[7] M. Izal, J. Aracil, D. Morató, and E. Magaña. Delay-throughput curves for timer-based OBS burstifiers with light load. *IEEE/OSA Journal of Lightwave Technology*, 2005.

[8] J. P. Jue and V. M. Vokkarane. *Optical Burst Switched Networks*. Springer, 2005.

[9] J. Kim, J. Cho, M. Jain, D. Gutierrez, L. Kazocsky, C. Su, R. Rabbat, and T. Hamada. Demonstration of a 2.5 Gbps Optical Burst Switched WDM Ring Network. In *Proceedings of OFC/NFOEC*, March 2006.

[10] D. Morató, J. Aracil, L. Díez, M. Izal, and E. Magaña. On linear prediction of internet traffic for packet and burst switching networks. In *Proceedings of the IEEE ICCCN'01*, 2001.

[11] Optical Internet Research Center. OIRC OBS-ns Simulator, April 2008. `http://wine.icu.ac.kr/~obsns/`.

[12] C. Qiao. Labeled optical burst switching for IP-over-WDM integration. *Communications Magazine, IEEE*, 38(9):104–114, Sep 2000.

[13] C. Qiao and M. Yoo. Optical Burst Switching (OBS) - A New Paradigm for an Optical Internet. *Journal of High-Speed Networks*, 8(1), 1999.

[14] M. Rodrigo and M. Remiche. Planning OBS Networks with QoS Constraints. *Photonic Network Communications*, 14(2):229–239, October 2007.

[15] Y. Sun, T. Hashiguchi, V. Minh, X. Wang, H. Morikawa, and T. Aoyama. A Burst-Switched Photonic Network Testbed: Its Architecture, Protocols and Experiments. *IEICE Transactions on Communicactions*, E88-B(10):3864–3873, October 2005.

[16] Y. Sun, T. Hashiguchi, V. Minh, X. Wang, H. Morikawa, and T. Aoyama. Design and Implementation of an Optical Burst-Switched Network Testbed. *IEEE Communications Magazine*, 43(11):S48–S55, November 2005.

[17] J. Teng and George N. Rouskas. A Comparison of the JIT, JET and Horizon Wavelength Reservation Schemes on a Single OBS Node. In *Proceedings of the First International Workshop on Optical Burst Switching (WOBS)*, Dallas, Texas, October 2003.

[18] J. S. Turner. Terabit burst switching. *J. High Speed Netw.*, 8(1):3–16, 1999.

[19] A. Varga. The OMNET++ discrete event simulation system. In *Proceedings of the European Simulation Multiconference*, pages 319–324, Prague, Czech Republic, June 2001. SCS – European Publishing House.

[20] A. Varga and R. Hornig. An overview of the OMNeT++ simulation environment. In *Simutools '08: Proceedings of the 1st international conference on Simulation tools and techniques for communications, networks and systems & workshops*, pages 1–10, ICST, Brussels, Belgium, Belgium, 2008. ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering).

[21] V. M. Vokkarane and J. P. Jue. Prioritized Burst Segmentation and Composite Burst-Assembly Techniques for QoS Support in Optical Burst-Switched Networks. *IEEE Journal on Selected Areas in Communications*, 21(7):1198–1209, September 2003.

[22] J. Wei and J. McFarland, R.I. Just-in-time signaling for WDM optical burst switching networks. *Lightwave Technology, Journal of*, 18(12):2019–2037, Dec 2000.

[23] S. Yao, B. Mukherjee, and S. Dixit. Advances in Photonic Packet Switching: An Overview. *IEEE Communications Magazine*, 38(2):84–94, February 2000.

[24] M. Yoo, C. Qiao, and S. Dixit. QoS performance of optical burst switching in IP-over-WDM networks. *Selected Areas in Communications, IEEE Journal on*, 18(10):2062–2071, Oct 2000.

[25] X. Yu, Y. Chen, and C. Qiao. A Study of Traffic Statistics of Assembled Burst Traffic in Optical Burst Switched Networks. In *Proceedings of SPIE Opticomm*, pages 149–159, Boston, July 2002. SPIE.

[26] A. Zalesky, E. Wong, M. Zukerman, H. L. Vu, and R. Tucker. Performance analysis of an OBS edge router. *Photonics Technology Letters, IEEE*, 16(2):695–697, Feb. 2004.

[27] W. Zhang, J. Wu, J. Li, W. Minxue, and S. Jindan. TCP Performance Experiment on LOBS Network Testbed. In *LNCS 4534, Proceedings of the 11th International IFIP TC6 Conference, ONDM*, pages 186–193, Athens, Greece, May 2007.

*Digital Object Identifier: 10.4108/ICST.SIMUTOOLS2010.8708*
*http://dx.doi.org/10.4108/ICST.SIMUTOOLS2010.8708*