Available online at www.sciencedirect.com
SCIENCE DIRECT®
Computer Communications 29 (2006) 865–878
computer communications
www.elsevier.com/locate/comcom

The prediction-based routing in optical transport networks

Eva Marín-Tordera*, Xavier Masip-Bruin, Sergio Sánchez-Lopez, Josep Solé-Pareta, Jordi Domingo-Pascual

Advanced Broadband Communications Center, Universitat Politècnica de Catalunya, Av. Víctor Balaguer s/n, Vilanova i la Geltru-Barcelona, 08800 Catalunya, Spain

Available online 12 September 2005

Abstract

In optical networks, the traditional routing problem is generally decoupled into two subproblems, the route selection and the wavelength assignment. Usual RWA (Routing and Wavelength Assignment) algorithms take the lightpath decision based on the network state information existing in those nodes caring about it. Unfortunately, this information might not be accurate enough so that the routing decisions could be incorrectly performed hence driving to a significant connection blocking increment. The mechanism proposed in this paper aims to optimize the network performance while reducing the signalling overhead required to keep updated network state information on all the network nodes. The novel idea introduced in the Prediction Based Routing allows lightpaths to be computed not according to the potentially inaccurate network state information but according to a prediction scheme. In this way, flooding update messages is not needed so that the signalling overhead is significantly reduced. After analyzing the PBR behaviour, we conclude that the PBR performs better than usual RWA algorithms (such as the Shortest-Path/Least-Loaded) in different scenarios of traffic loads and available resources.

Keywords: Optical routing; Routing inaccuracy; Prediction-based routing

1. Introduction

In recent years, the introduction of high capacity and reliable transport networks has become necessary in order to cover Internet traffic demands. New Internet applications increasingly request greater capacity and guarantees of traffic delivery in such a way that the traffic transmission model must be modified. 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. A wavelength routed *WDM* network is a circuit-switched network, in which a lightpath must be established between a source–destination node pair before data can be transferred. A lightpath is an end-to-end connection between a source–destination node pair, which may span multiple fibre links and use a single or multiple wavelengths. When the *OTN* includes automatic switching capabilities, it is referred to as an Automatically Switched Optical Network (*ASON*).

Source-based routing is one of the recommendations stated in the ASON specifications [1]. 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. Unlike traditional IP networks where the routing process only involves a physical path selection, in *OTN*s the routing process not only involves a physical path selection process (i.e. find a route from the source to the destination nodes) 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. There are many contributions in the literature addressing the *RWA* problem and proposing some algorithms dealing with both the path selection, and the wavelength assignment subproblems. On the one hand, the routing algorithms can be classified in two different classes: static and dynamic. In static routing, routes are

* Corresponding author. Tel.: +34 93 896 77 67; fax: +34 93 896 77 00.
E-mail address: eva@ac.upc.edu (E. Marín-Tordera).

0140-3664/$ - see front matter
doi:10.1016/j.comcom.2005.08.008

always precomputed and hence fixed for every source–destination node pair. 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 Ref. [2], authors propose the fixed-alternate routing algorithm which provides the network with more routes for each pair of nodes. Unfortunately, static routing does not consider the current network state when computing routes, which significantly impacts on the global network performance. Instead dynamic (or adaptive) routing relies on the network state information when computing routes. Just as an example, we can mention the Least-Loaded Routing (*LLR*) [3], where the selected route is the less congested among a set of predetermined routes, and the Weighted Least-Congestion Routing (*WLCR-FF*) [4] which simultaneously addresses both the path selection and the wavelength assignment problems.

On the other hand, there are several algorithms already proposed in the literature dealing with the wavelength assignment problem, such as Random, First-Fit, Least-Used, Most-Used, Min-Product, Least-Loaded, Max-Sum, Relative Capacity Loss, Protecting Threshold, and Distributed Capacity Loss. A significant collection of them can be found in Ref. [5].

In general, to establish a lightpath, that is select a route and 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, each Optical Cross-Connect (*OXC*) is 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 (for example [6,7]). Unfortunately, wavelength converters are still very expensive. There are many proposals to allow the network to include wavelength conversion capabilities also minimizing the economical cost [8].

As stated above, one of the ASON recommendations focuses on *RWA* solutions based on distributed source-routing. In this scenario, the routing inaccuracy problem comes up. The routing inaccuracy problem describes the impact on global network performance because of taking *RWA* decisions according to inaccurate or outdated routing information. In highly dynamic networks, inaccuracy is mainly due to the restriction of aggregating routing information in the update messages, the frequency of updating the network state databases and the latency associated with the flooding process. It has been clearly shown [9] that the routing inaccuracy problem may have a significant impact on global network performance in terms of connection blocking.

In this paper, we propose a routing mechanism based on prediction concepts, named the Prediction-Based Routing (*PBR*), aiming to reduce both the signaling 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 connections set-up. After comparing the results obtained by the *PBR* mechanism with a usual *RWA* algorithm (*Shortest-Path/Least-Loaded*) we conclude that the *PBR* can be a valid option to dynamically select lightpaths.

The remainder of this paper is organized as follows. Section 2 reviews main significant contributions existing in the recent literature dealing with the routing inaccuracy problem. Then, Section 3 describes in detail the routing mechanism proposed in this paper, also introducing the routing algorithm to be used to predict lightpaths availability as well as proposes an algorithm enhancement looking for simplicity and scalability. Section 4 presents an example to illustrate the *PBR* performance. In Section 5, we evaluate the *PBR* mechanism by simulation in different network scenarios and finally Section 6 concludes the paper.

2. Handling the routing inaccuracy problem

Most of the 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 (known as the routing inaccuracy problem). The most recent related work dealing with the routing inaccuracy problem is summarized in the following paragraphs.

In Ref. [9], the effects produced in the blocking probability because of having inaccurate routing information when selecting lightpaths are shown by simulation. The authors verify that the blocking ratio increases in a fixed topology when routing is done under inaccurate routing information. The routing uncertainty is introduced by adding an update interval of 10 s. Some other simulations are performed to show the effects on the blocking ratio because of changing the number of fibers on all the links. Finally, the authors argue that new *RWA* algorithms that can tolerate imprecise global network state information must be developed for dynamic connection management in *WDM* networks.

In Ref. [10], the routing inaccuracy problem is addressed by modifying the lightpath control mechanism, and a new distributed lightpath control based on destination routing is suggested. The mechanism is based on both selecting the physical route and wavelength on the destination node and adding rerouting capabilities to the intermediate nodes to avoid blocking a connection when the selected wavelength is no longer available at set-up time in any intermediate node along the lightpath. There are two main weaknesses of this mechanism. First, since the rerouting is performed in real time during the set-up process, wavelength usage deterioration is directly proportional to the number of intermediate nodes that must reroute the traffic. Second, the signalling overhead is not reduced, since the *RWA* 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 Ref. [11] where authors propose a mechanism aiming to control the amount of signalling messages flooded throughout network. Assuming that update messages are sent according to a hold-down timer fully regardless of the frequency of network state changes, authors propose a dynamic distributed *bucket-based Shared Path Protection* scheme. In this scheme, 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 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 increase in the connection blocking may be limited by properly introducing the suitable frequency of update messages. According to the simulation results obtained when applying the proposed scheme along with a modified version of the OSPF protocol, authors conclude that their proposal may help network operators to determine the frequency of update messages which best maintains a trade-off between the connection blocking and the signalling overhead.

In Ref. [12], authors propose a new adaptive source routing mechanism named *BYPASS Based Optical Routing* (*BBOR*), aiming to reduce the routing inaccuracy effects, i.e. blocking probability increment and non-optimal path selection, in WS networks. In Ref. [8], authors extend the mechanism to be applied to networks with conversion capabilities. The BBOR mechanism is based on bypassing those links, which cannot forward the setup message because they lack the selected wavelength. The bypass is achieved by forwarding the setup messages through a previously precomputed alternative path, named bypass-path.

3. The prediction-based routing mechanism

In this section, we present a thorough description of the *Prediction Based Routing (PBR)* mechanism. The main idea of the *PBR* mechanism is based on extending the concepts of branch prediction used in the computer architecture area [13]. 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. Bringing this concept to a network scenario, we state that 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 set-up. 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. Notice that a minimal updating is required to ensure connectivity.

Summarizing, 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 path. Sections 3.1–3.4 clearly describe the *PBR* mechanism.

3.1. History registration

Assuming source routing, all the source nodes must have a *history*, which basically includes the information about wavelengths and links they have previously assigned. This history is repeated all through the time and is stored in a history register named *Wavelength Register* (*WR*) which will be used as a pattern of behaviour. The *WRs* hold different vectors of 0 and 1 s reflecting the history. There is one of such registers for every wavelength on every path to every destination node. Before describing the *PBR* mechanism it is necessary to note that the *WR*s are modified every unit of time. We define a unit of time as the value we use to measure the simulations timing, including holding time, arrival time, and time between updating. In our simulations, we fix the arrival time to 10 units of time; thus, the rest of values are fixed according to this one.

The method used to register the history of the network state in every source node is based on assuming that for each unit of time, each *WR* is updated with a 0 value whenever this wavelength on this path is used on that unit of time. Otherwise, the register of an unused wavelength on this path is updated with 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.

3.2. 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 by means of a counter. One *PT* is needed in the source nodes for every feasible route between any source–destination node pair. For example, assuming that a source node sends traffic towards two different destination nodes where every source–destination node pair has two different routes (assuming for instance the two shortest-paths) with six 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 the 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 the 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 [13]. As presented in Ref. [13] 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 Ref. [13] 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.

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 route and wavelength is selected but the connection request has been 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.

3.3. Routing algorithm

Based on the *PBR* mechanism, we define a new *RWA* prediction algorithm, named *Route and Wavelength Prediction (RWP) algorithm*, 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. We assume 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. We assume in our evaluation that SP_1 and SP_2 are link disjoint because the prediction is to use a completely different route (since the source node does not know the link blocking the route, if any) whenever the first one is predicted to be blocked. The *PT*s are accessed by one index per table which is built from the corresponding *WR*, obtaining the two-bit counters values. In Fig. 1, we present a flow chart depicting the *RWP* performance assuming *U* wavelengths in every link. The *RWP* algorithm always starts

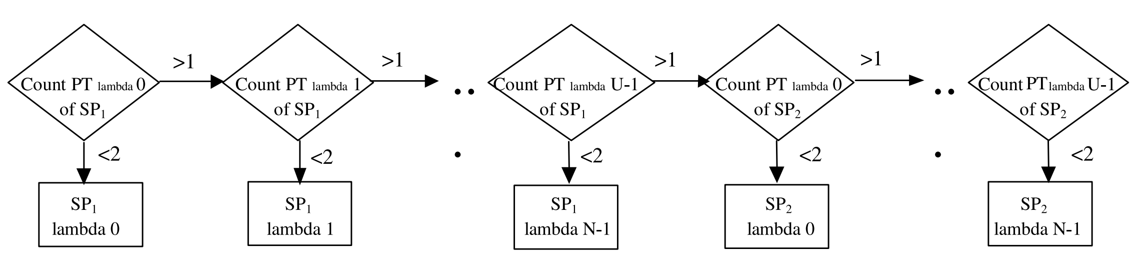

Fig. 1. *RWP* flow chart.

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 less 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 2 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

```
1.     Order(Route SP1)
           (o0, o1......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(oi)<2 and wavelength oi is available in output link to route SP1)
                       { assign route SP1 and wavelength oi;
                               if connection is established decrease PTcounter(oi)
                               else increase PTcounter(oi)
        }endif
       i++;
}endwhile
3.     If (route is not assigned) {
4.     Order(Route SP2)
           (o0, o1......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(oi)<2 and wavelength oi is available in output link to route SP2 )
                               { assign route SP2 and wavelength oi;
                                       if connection can be established decrease PTcounter(oi)
                                       else increase PTcounter(oi)
                       }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
```

Fig. 2. Pseudo-code of the *RWP*-o algorithm.

towards one of the two shortest paths. The attempt of trying to select the routes by just checking the output availability when no lightpath can be assigned is done to unblock the PT counters. Indeed, when no path and wavelength is selected (all PT counters are bigger than 1), the *PBR* algorithm assigns the request to the first available wavelength on the ouput link towards the SP_1. But, if even so the path cannot be assigned, then the algorithm assigns it to the first available wavelength on the output link towards the SP_2. If the path and wavelength can be selected by means of this second 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 Section 3.1, 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*.

Up to now, we have assumed one fibre per link in the *RWP* description. However, the algorithm must be enhanced when assuming *n* possible fibres. Although the algorithm always checks SP_1 and SP_2 in this order, we allow the algorithm to 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, we name the algorithm *RWP-f*. 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*. 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, 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 Fig. 2, we show 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.

3.4. Routing algorithm enhancement

The algorithm enhancement introduced in this paper, 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 need of registering information about the last and previous units of time makes the *PBR* mechanism regardless of the unit of time selection. We justify this enhancement by means of several simulations. The two-bit counter runs as follows: the value of the counter for a route and wavelength is 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 a blocking occurs 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 two different destination nodes through two possible routes for every destination, SP_1 and SP_2, and four possible wavelengths, then there are 16 two-bit counters in the source node. We name these two-bit counters as Wavelength-Route Counters, *WR*C. The enhanced algorithm runs as shown in Fig. 2 (notice that the *PT*s are only two-bit counters). Summarising, 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 (as 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.

4. Illustrative example

Before evaluating our proposal, we present an example of the enhanced algorithm to illustrate its performance. Fig. 3 shows the example topology being n1 is a source node and n2, n3 and n4 destination nodes. Moreover, we consider that a link consists of one fibre with two wavelengths. 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: lambda 1) WRC12AL2, WRC12BL1, WRC12BL2, WRC13ALl, WRC13AL2, WRC14AL1, WRC14AL2, WRC14BL1, WRC14BL2. We describe in detail below the evolution of the connection requests during 6 units of time.

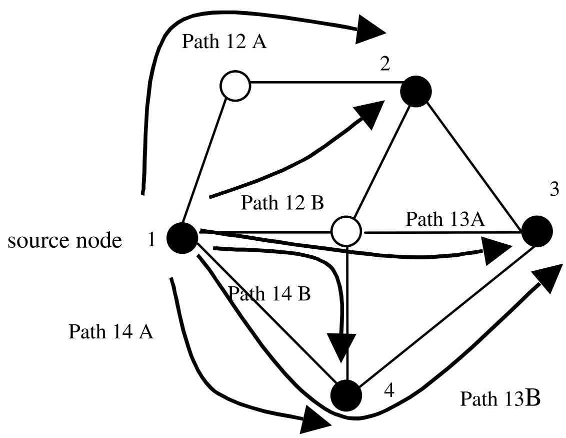

Fig. 3. Topology used in the illustrative example.

Unit of time 1: A new connection request between nodes 1 and 4 reaches n1 with a holding time of 4 units of time. We suppose that no more connections are established between node 1 and any destination. Fig. 4(a) shows both how the counters are read and how the prediction process works. Suppose that the algorithm order 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 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 therefore this route and wavelength are 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 Fig. 4(b), we can see the updating of the *WRC* 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 nodes 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 lambda 1 is predicted to be available but the connection request is blocked. Fig. 5(a) shows the prediction process. The counter WRC12AL1 is immediately updated hence being increasing (Fig. 5(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 frees its links because the holding time has finished.

Unit of time 6: In this unit of time there are not new connection requests. The request between nodes 1 and 2 produced in unit of time 3 frees its links because the holding time has finished.

5. Performance evaluation

Once the proposed algorithm has been analyzed by the illustrative example presented in Section 4, we evaluate the performance of the *PBR* mechanism in two different network scenarios. First, we have carried out a preliminary evaluation of the *PBR* behavior, analyzing the effect of

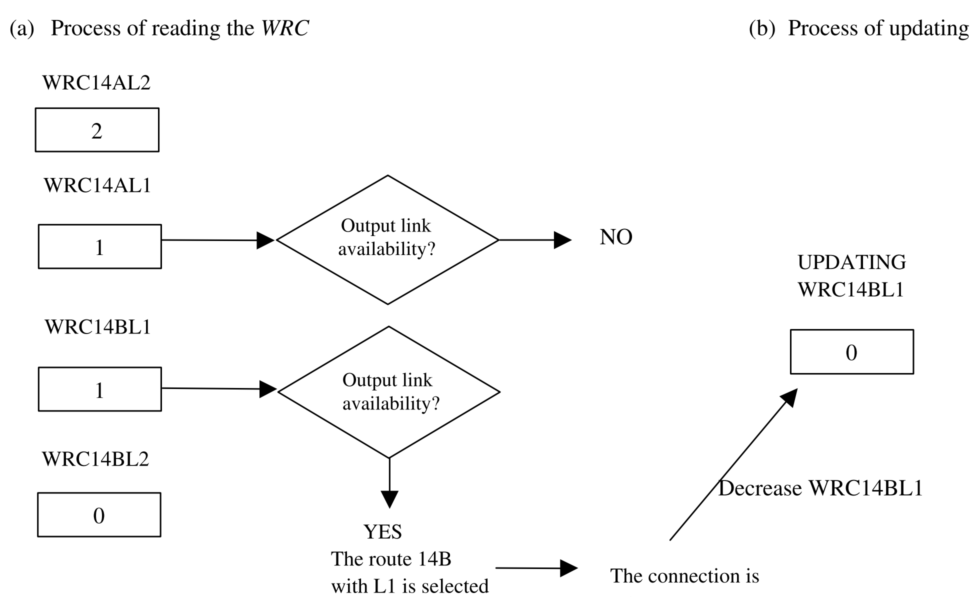

Fig. 4. Process of predicting the request between nodes n1 and n4 on the unit of time 1.

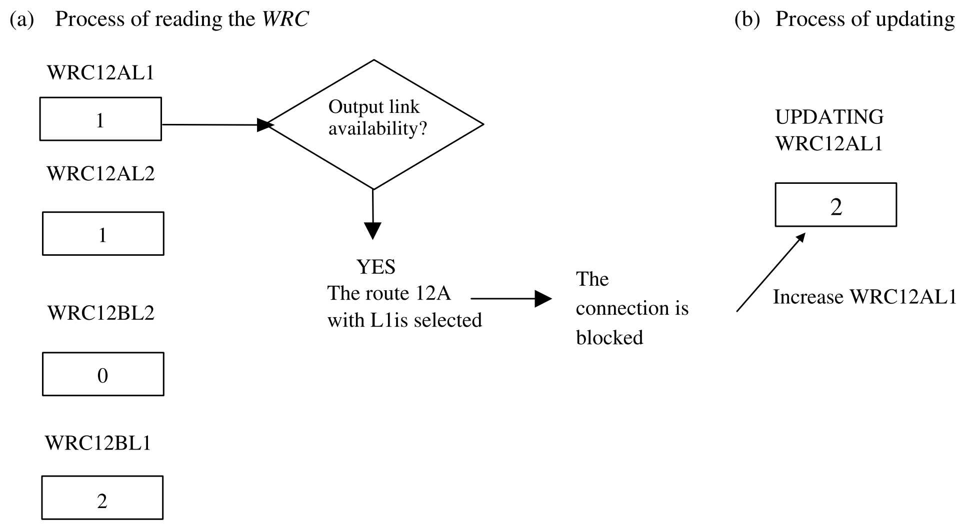

Fig. 5. Process of predicting the request between nodes n1 and n2 on the unit of time 2.

different parameters, such as the number of *WR*s bits or the number of fibres and wavelengths. We compare both, the *RWP-f* and the *RWP-o* with a well-known wavelength assignment algorithm, the Least-Loaded. In the second set of simulations, we compare the performance of the *PBR* mechanism with the Least-Loaded algorithm running on the Spanish National Research Network (RedIris) network topology.

5.1. Preliminary evaluation

We have carried out some simulations on the network topology shown in Fig. 6 that consists of 9 nodes where 2 of them are source nodes and other 2 destination nodes. However, unlike the illustrative example described in Section 4, in this case, we assume that 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.

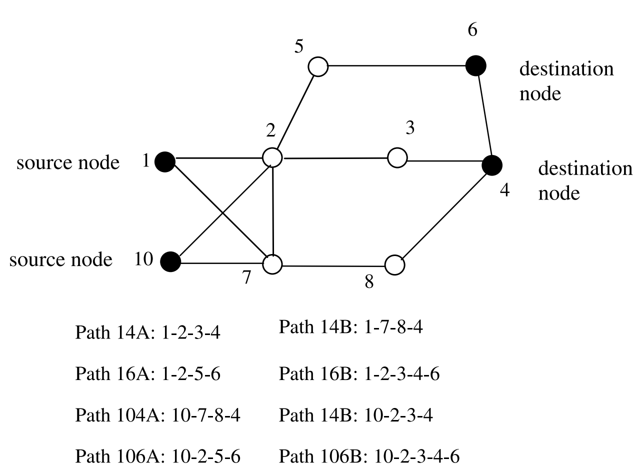

Fig. 6. Topology used in preliminary evaluation.

5.1.1. Number of WRs bits, RWP-f versus RWP-o

As mentioned in previous sections, we have proposed an enhancement of the *PBR* mechanism to reduce the algorithm complexity and scalability. To evaluate this proposal, we measure the effect of varying the number of *WR*s bits in the ratio of blocking. Simulations are obtained by applying the *PBR* to the topology of the Figs. 6–8 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 Fig. 6 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, we conclude that the optimal number of bits depends on different parameters such as the traffic load, number of wavelengths (lambdas in figures) and fibres. Just as an example, in Fig. 7(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*s. 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. In general, the performance for 0 bits of *WR* is enough good, being in a lot of cases the best. For this reason, we have decided to enhance the algorithm, in such a way that *PT*s are only of one entry (i.e. only one two-bit counter per route and wavelength).

On the other hand, if we compare the results for the two options of ordering the checking of 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. Due to the two reasons exposed above from now on we only present results for the *RWP*-o algorithm without *WR*s.

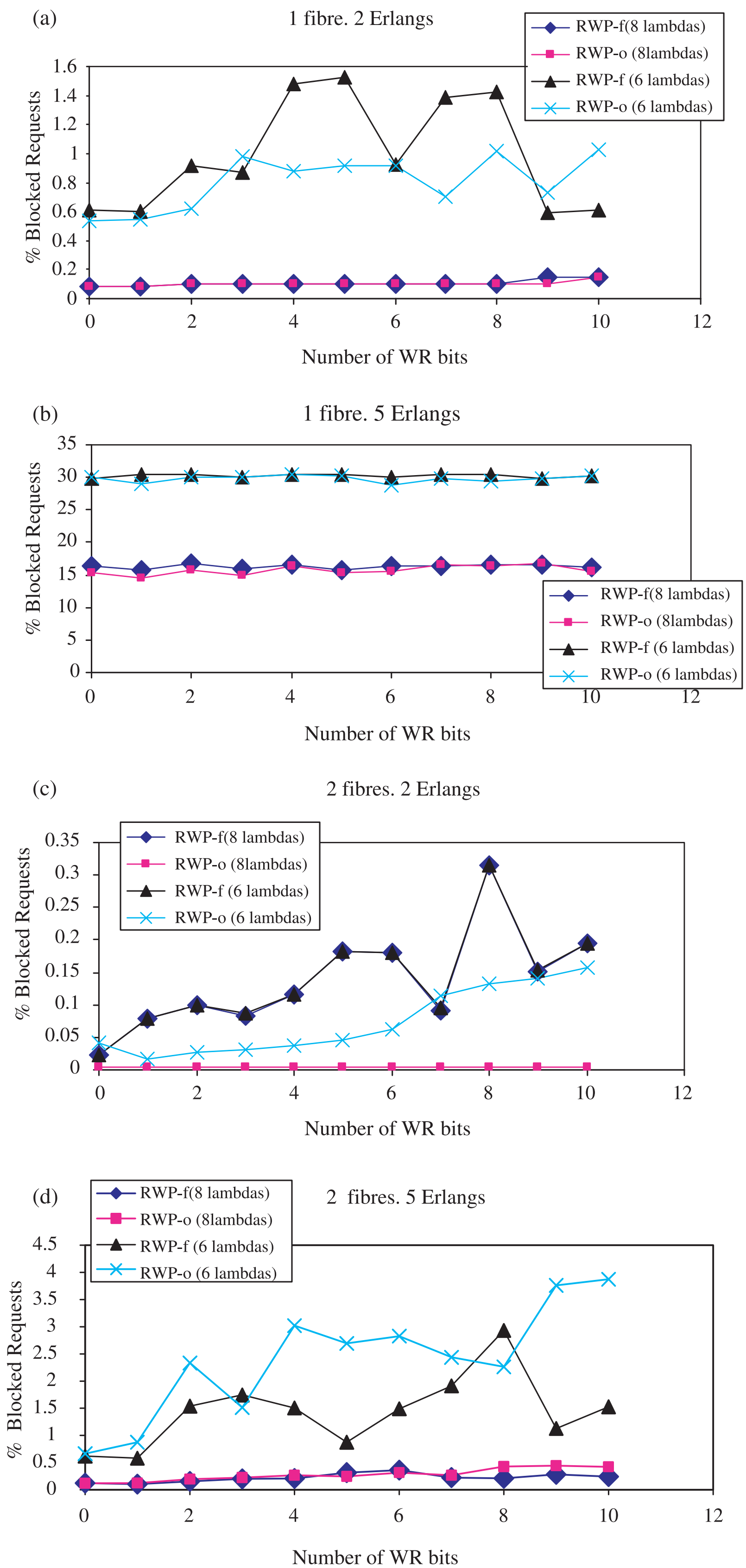

Fig. 7. Percentage of blocked connection versus number of *WR* bits for *RWP*-f and *RWP*-o algorithms (1 and 2 fibres).

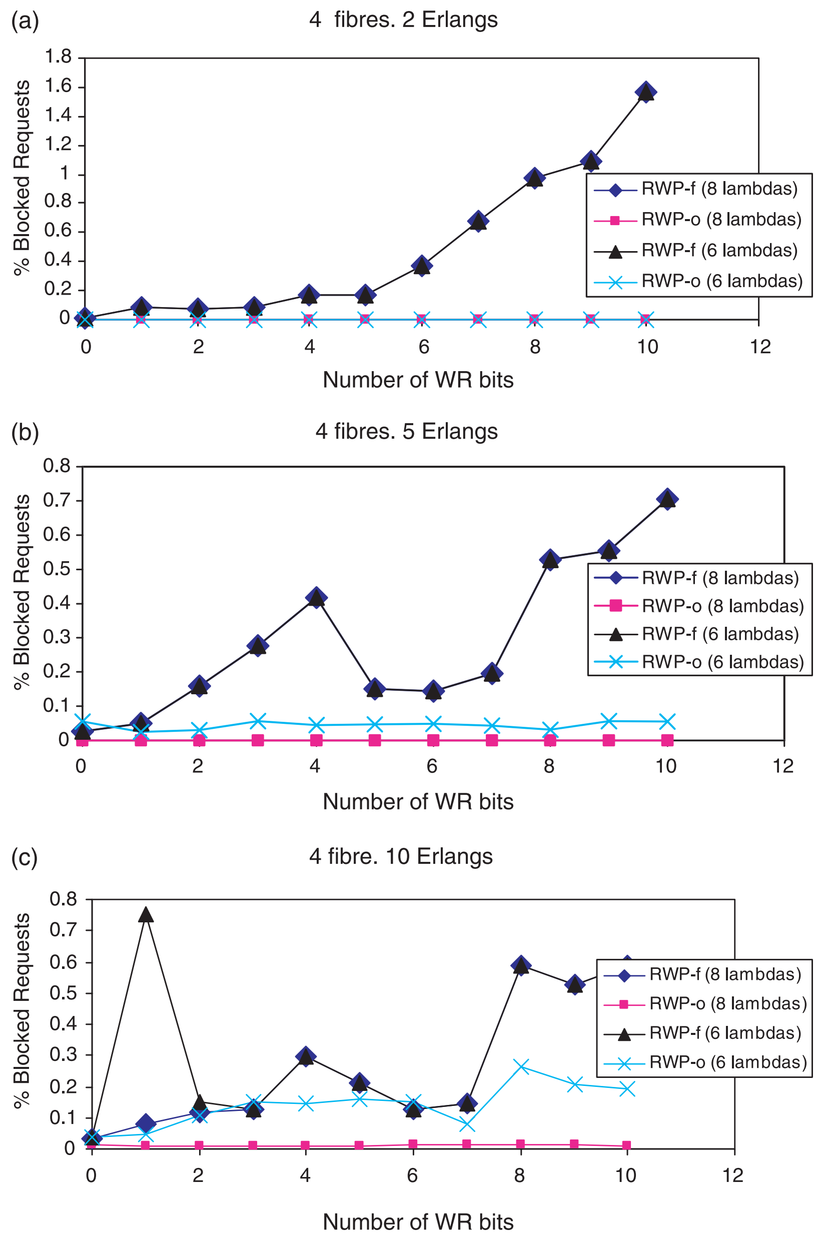

Fig. 8. Percentage of blocked connection versus number of *WR* bits for *RWP*-f and *RWP*-o algorithms (4 fibres).

5.1.2. RWP-o versus Least-loaded

After evaluating the enhancement of the algorithm, we compare the performance of the *RWP-o* algorithm with the performance of a well-known and commonly used route and wavelength assignment mechanism, Shortest-Path combined with the Least-Loaded. The Least-Loaded algorithm selects the wavelength that has the largest residual capacity (number of free fibres for that wavelength), on the most loaded link along the route. Note that in single-fibre networks the *Least-Loaded* becomes the *First-Fit* algorithm. Both heuristic algorithms, *First-Fit* and *Least-Loaded*, need network state update messages to know the wavelength availability along the route. On the other hand, the *PBR* mechanism only utilizes the information about output link availability, local information about wavelength availability along the route and the information contained in the *PTs*. We have implemented the same two shortest and link-disjoint paths for the *Least-Loaded* algorithm. The *Least-Loaded* algorithm checks first the wavelengths of the first path, SP_1, and it then chooses the wavelength with more availability (more free fibres). But if there is not wavelength availability in path SP_1, it checks the wavelengths of path SP_2, also selecting the more available.

We have carried out a set of simulations in the topology of Fig. 6, varying the time between the updating (units of time), and the results are presented in Fig. 9 (1 and 2 fibres, for 2 and 5 Erlangs) and Fig. 10 (4 fibres for 5 and 10 Erlangs). In Fig. 9, we only present results for 2 and 5

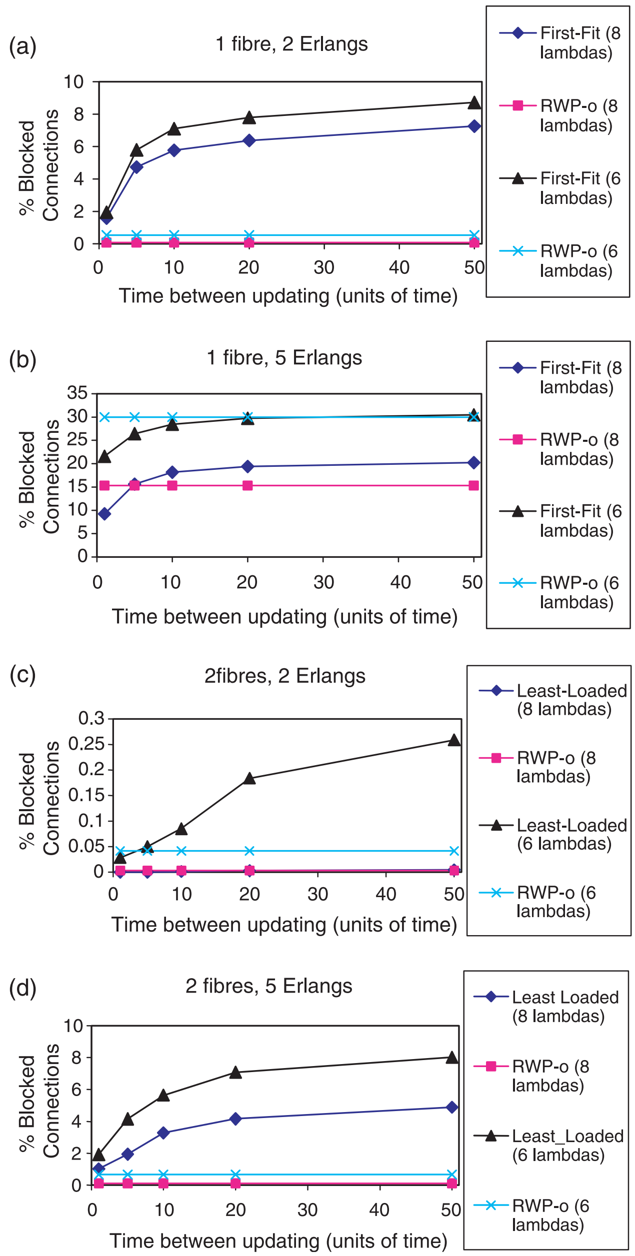

Fig. 9. *RWP-o* versus First-Fit (1 fibre) and versus Least-Loaded (2 fibres).

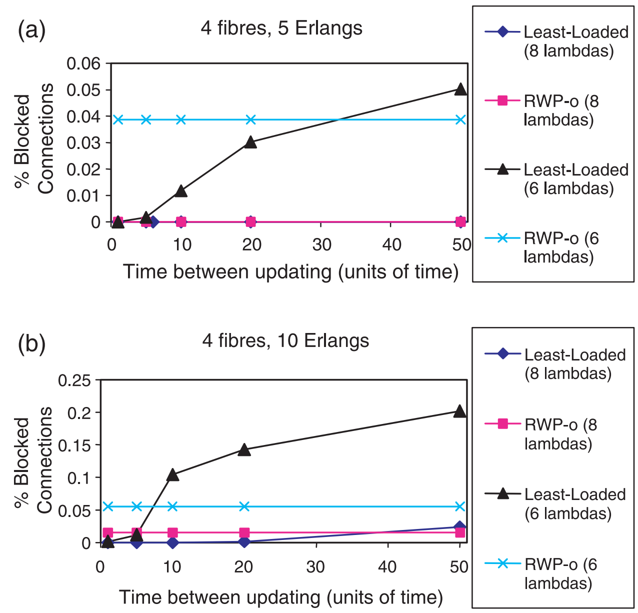

Fig. 10. *RWP-o* versus Least-Loaded for 4 fibres.

Erlangs because the percentage of blocked connections for 10 Erlangs is very high for both algorithms. On the other hand, in Fig. 10 (4 fibres) we show the results for 5 and 10 Erlangs because for 2 Erlangs for both algorithms and for the range of updating values, the number of blocked connections is 0. Notice that in Figs. 9 and 10 the *RWP*-o algorithm does not vary with the time between updates because it does not need network state update messages.

In Fig. 9(a), the results obtained for 1 fibres, 2 Erlangs and 6 or 8 lambdas depict that the *RWP*-o algorithm outperforms the *First-Fit* algorithm (= *Least-Loaded* with 1 fibre), even in the ideal case where the update messages are flooded every unit of time. For 5 Erlangs (Fig. 9(b)) and 8 lambdas the *RWP*-o algorithm has similar results than the *First-Fit* algorithm with updating every 5 units of time. But for 6 lambdas and 5 Erlangs, the *RWP*-o algorithm only has similar performance than the *Least-Loaded* 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 Fig. 9(c) and (d). For 2 Erlangs and 8 lambdas both algorithms, *RWP-o* and *Least-Loaded*, have a blocking percentage practically equal to 0. However, for 6 lambdas the *RWP-o* algorithm has similar performance than the *Least-Loaded* with updating every 5 units of time. On the other for 5 Erlangs (Fig. 9(d)) the *RWP-o* algorithm outperforms the *Least-Loaded* algorithm even on the ideal case with updating every unit of time.

Results in Fig. 10 correspond to simulations carried out with 4 fibres. For 5 Erlangs (Fig. 10(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 0.06%, and the *RWP-o* algorithm has similar results than the *Least-Loaded* with updating between 10 and 20 units of time.

For 10 Erlangs (Fig. 10(b)) and 8 lambdas the *RWP-o* algorithm has similar performance than the *Least-Loaded* with updating every 5 units of time. We also observe that for 6 lambdas the *RWP-o* algorithm cross the results of the *Least-Loaded* algorithm when the updating is between 5 and 10 units of time.

Summarizing, we observe that the *RWP-o* algorithm outperforms the *Least-Loaded* 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 (Fig. 9(b)) the

Least-Loaded 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 *Least-Loaded* also outperforms the *RWP-o* algorithm (Fig. 10(a)). 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 significant signalling overhead reduction.

5.1.3. Capability of learning

Finally, we evaluate the time required to train the *WR*C, i.e. the learning time for the *RWP-o* algorithm. Results on Fig. 11 show the number of blocked connections versus the number of connections requests for the *RWP-o* algorithm and the *Least-Loaded* (updating 1 and 5) for the first 5000 connection requests. We analyze the warm-up phase to check how the two-bit counters are trained. Results are shown for 8 lambdas, 5 Erlangs and with 1 and 2 fibres. We can see from Fig. 11(a) (2 fibres) that initially the *RWP-o* algorithm fails more when assigning route and wavelength. Instead, after a training period, when the number of connection requests is 2500, the *RWP-o* algorithm outperforms the *Least-Loaded* algorithm when the updating is every 5 units of time. However, when the number of connection requests is more or less 3700 the *RWP-o* outperforms the *Least-Loaded* results when the updating is every unit of time. For this traffic load, we can consider that the *RWP-o* has a 'time of learning' of 2500 or 3700 request, depending on what we define as learning time. Moreover, we present results for 1 fibre on the same scenario of simulation. In this case, the *RWP-o* has similar number of

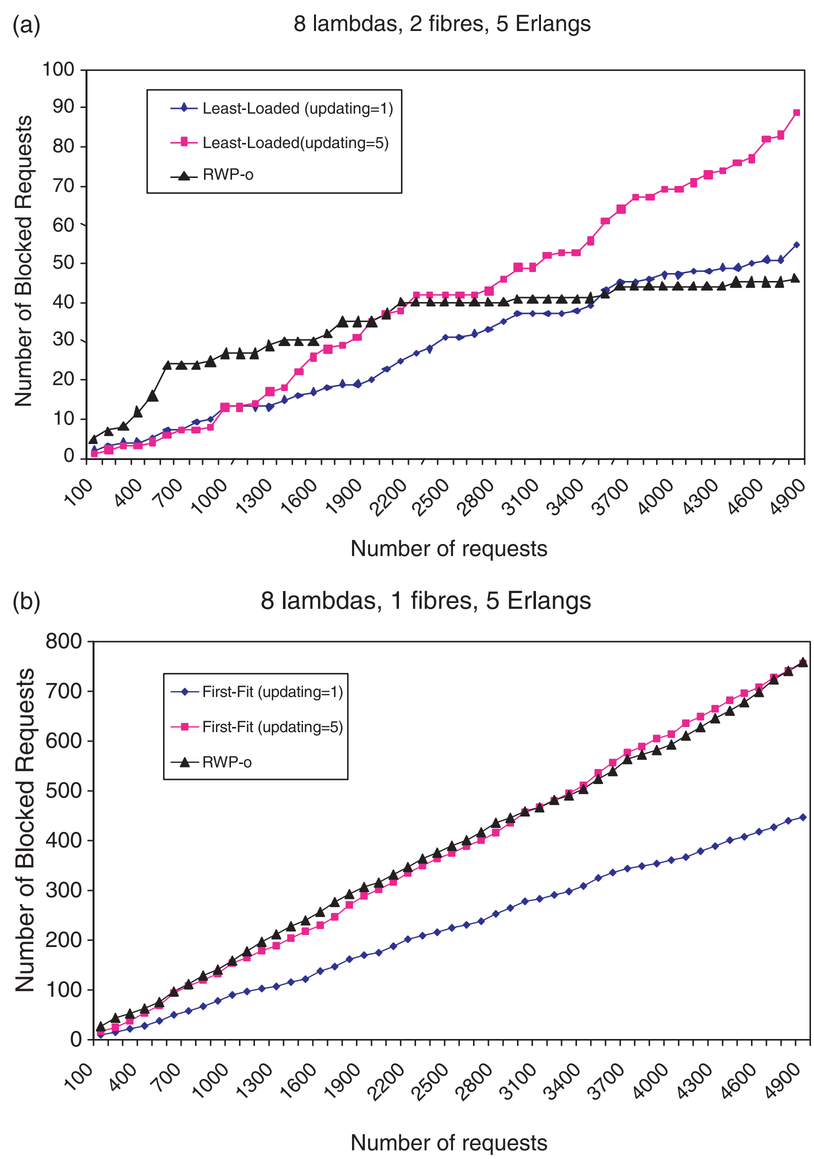

Fig. 11. Evolution in the number of blocked connection for the *RWP*-o algorithm and the Firs-Fit/Least-Loaded (updating = 1 and 5) for the first 5000 connection requests.

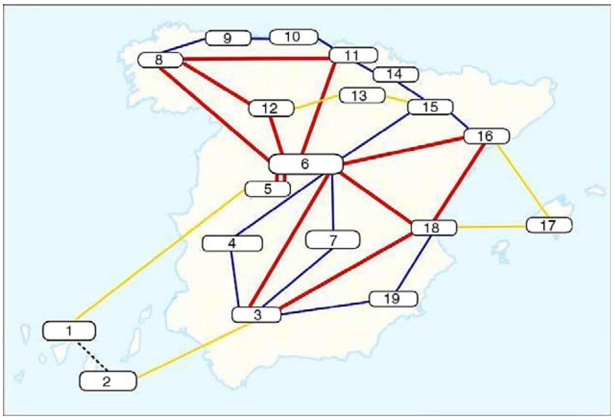

Fig. 12. Spanish national research network (RedIris) topology.

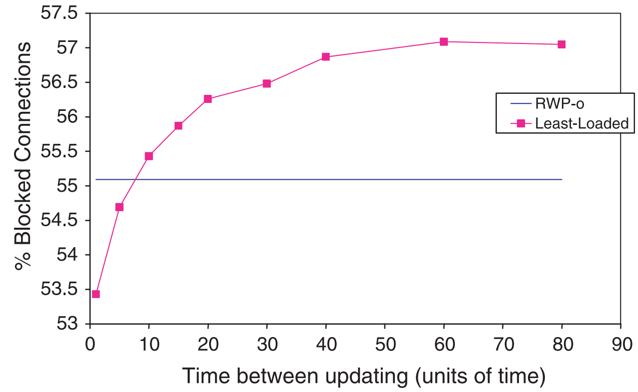

Fig. 14. % of Blocked Connection of the *RWP-o* and Least-Loaded algorithms versus time between updating.

blocked connections than the *Least-Loaded* algorithm when the updating is every 5 units of time. Therefore, we can argue that the 'time of learning' of the *RWP-o* algorithm strongly depends on the traffic pattern, and the available resources.

5.2. Results in the Spanish national research network (RedIris)

We have carried out a set of simulations in the topology of the Spanish National Research Network (RedIris) shown in Fig. 12. The simulation environment consists of the following features: 7 fibres per link; 4 wavelengths per fibre; nodes 1 and 2 are considered as source nodes while nodes 15 and 16 are destinations nodes; a Poisson distribution models connection arrivals on the wavelength switching network.

5.2.1. Results varying the traffic load

In Fig. 13, we present the results in percentage of blocked connections when the traffic load ranges from 1 to 20 Erlangs, for the *RWP-o* and the *Least-Loaded* algorithms in an ideal case, i.e. updating every unit of time, and when updating every 20 units of time. It is important to note that one update message every single unit of time is prohibitive due to the huge signalling overhead.

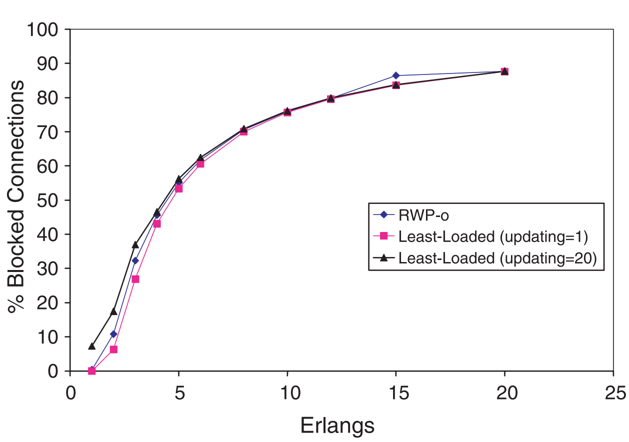

Fig. 13. % of Blocked Connection of the *RWP-o* and Least-Loaded algorithm for versus the traffic load.

From Fig. 13, we observe that for light traffic load lower than 5 Erlangs the *RWP-o* algorithm has similar performance that the *Least-Loaded* algorithm in the ideal case, i.e. updating every cycle. On the other hand, for medium traffic loads (from 5 to 15 Erlangs) the results of the *RWP-o* algorithm are very close to the results of the *Least-Loaded* algorithm assuming updating every 20 cycles. Finally, for high traffic load (> 15 Erlangs) there is not more differences between the results of both algorithms because the network is overloaded.

5.2.2. Results varying the time between updating

Finally, we present results in Fig. 14 of the *RWP-o* and the *Least-Loaded* algorithms in percentage of blocked connections when the time of the updating process varies. We keep the same simulations conditions than those used in the previous simulations (Section 5.2.1) but fixing the traffic load to 5 Erlangs. We can see how the percentage of blocked connection produced by the *Least-Loaded* algorithm increases when the updating time is also increased. That is, the performance of usual *RWA* algorithm (*Shortest-Path/Least-Loaded*) depends on the time the network state database is updated. Flooding update messages every one unit, or even every 5 units, of time is unaffordable from the point of view of the signalling overhead. In Fig. 14, we can see that the results of the *RWP-o* algorithm reach the results of the *Least-Loaded* algorithm when the updating time is between 5 and 10 units of time.

6. Conclusions

In this paper, authors propose the Prediction-Based Routing (*PBR*) mechanism to tackle the *RWA* problem in Optical Transport Networks. The main characteristic of the *PBR* is to provide source nodes with the capability of taking routing decisions without using the network state information contained in their network state databases. Two immediate benefits may be inferred from the *PBR* mechanism. The former, the *PBR* removes the update messages required to update the network state databases (only connectivity messages are required). The latter, the *PBR* reduces the blocking produced by the routing inaccuracy problem. It is worth mentioning that the fact of removing the update messages flooding not only does not make blocking probability worse but instead exhibits even lower blocking results. The *PBR* has been evaluated by simulating two different network topologies under several traffic conditions. After analyzing the obtained results, we can conclude that the *PBR* can be proposed as a good option to reduce the signalling overhead in an *OTN* without affecting on the global network performance.

Acknowledgements

This work was partially funded by the MCyT (Spanish Ministry of Science and Technology) under contract FEDER-TIC2002-04344-C02-02 and the CIRIT (Catalan Research Council) under contract 2001-SGR00226.

References

[1] ITU-T Rec.G8080/Y.1304, Architecture for the Automatically Switched Optical Network (ASON), November 2001.

[2] H. Harari, M. Massayuki, H. Miyahara, Performance of alternate routing methods in all-optical networks, IEEE INFOCOM'96 1996.

[3] K.M. Chan, T.S.P. Yun, Least congested path routing methods in WDM lightwave networks, Proceedings of IEEE INFOCOM'94 1994.

[4] B. Li, X. Chu, Routing and wavelength assignment vs. wavelength converter placement in all-optical networks, IEEE Optical Communications (2003).

[5] H. Zang, J. Pue, B. Mukherjee, A review of routing and wavelength assignment approaches for wavelength-routed optical WDM networks, Optical Network Magazine (2000).

[6] M. Kovacevic, A. Acampora, Benefits of wavelength translation in all-optical clear-channel networks, IEEE Journal on Selected Areas in Communications 14 (5) (1996).

[7] B. Ramamurthy, B. Mukherjee, Wavelength conversion in WDM networking, IEEE JSAC 16 (7) (1998).

[8] 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, Proceedings of IEEE GLOBECOM 2003, San Francisco, USA 2003.

[9] H. Zang, X. Yuan, A study of dynamic routing and wavelength assignment with imprecise network state information, Proceedings of ICPP Workshop on Optical Networks (2002).

[10] J. Zheng, H. Mouftah, Distributed lightpath control based on destination routing in wavelength-routed WDM networks, Optical Networks Magazine 3 (4) (2002) 38–46.

[11] 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, Proceedings of ONDM'03, Budapest, Hungary 2003 pp. 433–447.

[12] 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, Proceedings of ONDM'03, Budapest, Hungary 2003.

[13] J.E. Smith, A study of branch prediction strategies, Proceedings of 8th International Symposium in Computer Architecture, Minneapolis 1981.