



ELSEVIER

Available at  
www.ComputerScienceWeb.com  
POWERED BY SCIENCE @ DIRECT®

COMPUTER  
NETWORKS

Computer Networks 41 (2003) 727–742

www.elsevier.com/locate/comnet

# Scheduling algorithms for multicast traffic in TDM/WDM networks with arbitrary tuning latencies <sup>☆</sup>

A. Bianco <sup>\*</sup>, G. Galante, E. Leonardi, F. Neri, A. Nucci

*Dipartimento di Elettronica, Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy*

Received 14 May 2002; received in revised form 7 October 2002; accepted 10 November 2002

Responsible Editor: G.N. Rouskas

## Abstract

We consider all-optical Time Division Multiplexing (TDM)/Wavelength Division Multiplexing (WDM) broadcast and select networks with slotted operation. Each network access node is equipped with one fixed transmitter and one tunable receiver; tuning times are not negligible with respect to the fixed size slot time. We discuss efficient scheduling algorithms to assign TDM/WDM slots to multicast traffic in such networks. The problem is shown to be NP-hard; thus, heuristic algorithms based on the Tabu Search meta-heuristic are proposed, and their performance are assessed using randomly created request matrices based on two types of multicast traffic patterns. We show that significant advantages can be obtained by using these novel algorithms with respect to simpler greedy algorithms, even when restricting CPU times to realistic values to make the algorithms of practical use.

© 2002 Elsevier Science B.V. All rights reserved.

*Keywords:* Wavelength division multiplexing; Time division multiplexing; Scheduling; Multicast traffic; Broadcast and select networks

## 1. Introduction

Broadcast-and-select networks usually exploit Wavelength Division Multiplexing (WDM) as well as Time Division Multiplexing (TDM) techniques for the provision of integrated services over a packet-mode transport. These systems, in which

each receiver has access to the optical signals generated by all transmitters, can offer transparent support to a large number of simultaneous very-high-speed end-to-end communications, provided that adequate scheduling algorithms are defined to dynamically share the available network resources.

We consider all-optical TDM/WDM broadcast-and-select networks in which  $N$  nodes communicate via  $W$  wavelengths. Node interfaces are equipped with *one* full-duplex transceiver; hence, each node can be source and destination of at most one data flow at any given time. Transmitters operate on fixed wavelengths, while receivers can be tuned to all available wavelengths. By tuning the receivers, and by dynamically allocating the  $W$  available wavelengths, full connectivity is achieved

<sup>☆</sup> A preliminary version of this paper was presented at the IEEE GLOBECOM conference in San Antonio, November 2001 [1].

<sup>\*</sup> Corresponding author. Tel.: +39-0-11-5644098; fax: +39-0-11-5644099.

*E-mail addresses:* [andrea.bianco@polito.it](mailto:andrea.bianco@polito.it) (A. Bianco), [galante@polito.it](mailto:galante@polito.it) (G. Galante), [leonardi@polito.it](mailto:leonardi@polito.it) (E. Leonardi), [neri@polito.it](mailto:neri@polito.it) (F. Neri), [nucci@polito.it](mailto:nucci@polito.it) (A. Nucci).

among nodes. WDM wavelengths are assumed to be slotted and synchronized; each fixed size slot can accommodate the transmission of one packet on one wavelength. Obviously, variable size packets must be segmented in fixed size packets that fit into one slot.

We do not deal with issues related with physical topology; thus, our algorithms can be applied to traditional broadcast and select access LAN based on star topologies, to LAN or MAN based on ring topologies [2,3], to nation-wide network topologies based on a single passive AWG device interconnecting PONs [4,5], and to MAN based on interconnected rings [6]. However, the proposed algorithms are not directly usable for general meshed topologies, since we disregard the problem of mapping logical topologies over a given physical topology. Nevertheless, they may be useful in several contexts, running, as outlined, from LAN through MAN to WAN networks.

The time required to tune transceivers is assumed to be *not* negligible with respect to the packet transmission time. This assumption is based on the observation that, with some of the optical components available today (e.g., those based on acousto-optics), the tuning latency can be quite larger (order of few  $\mu\text{s}$ ) than the packet transmission time (order of fractions of  $\mu\text{s}$ ). Note that small slot size are mandatory if the network wants to provide support to interactive voice applications, like traditional phone calls. Moreover, given the continuously increasing transmission speeds, even the mapping of maximum size IP packets may require slots of reasonably small size.

Our scheduling algorithms react to slowly-varying bandwidth requests: their computational complexity does not permit to react to node reservations on a packet-by-packet or connection-by-connection basis. Thus, the proposed algorithms are operating mainly at a provisioning level, e.g., as a new ISP is added to the network and an SLA is established, or a bandwidth increase in a logical link between two neighboring routers is required. We shall assume to allocate resources on the basis of a traffic request matrix, which describes nodes' transmission requirements, i.e., aggregate users' traffic requirements, according to traffic estimates derived either from forecasts, measurements or

commercial agreements. This scenario is normally called “off-line” scheduling, in contrast with “on-line” scheduling, where each request is served in the same order in which it is issued, i.e., quickly reacting to single users' or nodes' requests.

Several scheduling algorithms for unicast traffic have been proposed in the literature [7–13], but to the best of our knowledge only few works, e.g., [14–17], and [18], address the problem of the definition of efficient scheduling algorithms in presence of multicast traffic.

In [14] and [15] multicast scheduling algorithms were proposed and analyzed: multicast packets are transmitted in an atomic fashion.

In [16] and [17] it was shown that significant advantages can be achieved by partially transmitting multicast packets to sub-groups of destinations in different time slots. The scheduling algorithms proposed in [16] were conceived to regulate the access to shared wavelengths on a packet-by-packet basis. As a consequence, they are very simple, but their performance may become poor in certain traffic scenarios.

In [17] some heuristic algorithms to solve the multicast scheduling problem in the off-line context have been proposed.

In [18] the problem of minimizing the number of multicast transmissions in single-hop WDM networks with delay constraints is discussed; requests are served on a packet-by-packet basis, i.e., with reference to an on-line scenario.

The paper is organized as follows: in Section 2 we present the scheduling problem for both unicast and multicast traffic and give some lower bounds on the required frame length. In Section 3 we provide an Integer Linear Programming (ILP) formulation of the multicast scheduling problem. Since the ILP model cannot be solved in a reasonable time for networks having more than four nodes, four wavelengths and six multicast groups, in Section 4 we devise some heuristic approaches based on the Tabu Search [19] and discuss their computational complexity. Finally, in Section 5, after showing the optimal schedule for the largest instance we were able to solve on a 1 GHz PC, we analyze the performance benefits in terms of frame length and running times for different network sizes in several traffic scenarios. We show

that our family of algorithms provides a good trade-off between performance gain and algorithmic complexity. Significant performance gains are obtained with respect to traditional greedy approaches, and for several traffic scenarios our scheduling algorithms obtain a frame length very close to the optimum value. It is also shown that, when larger networks are considered, the algorithms can obtain a frame length very close to the optimal frame length with reasonably small CPU time.

## 2. Lower bounds on the frame length

We consider a network with  $N$  nodes and  $W$  wavelengths ( $W \leq N$ ), and denote with  $\mathcal{N} = \{1, 2, \dots, N\}$  the set of all nodes and with  $\mathcal{W} = \{1, 2, \dots, W\}$  the set of all wavelengths.

Node interfaces are equipped with one fixed transmitter and one tunable receiver; the tuning latency is denoted by  $T$  and is assumed to be not negligible with respect to the packet transmission time.

When  $W = N$ , source nodes correspond bi-univocally to the wavelengths on which data must be transmitted; if  $W < N$ , more transmitters share the same wavelength.

The transmitter wavelength assignment is fixed and independent of the traffic matrix, therefore the scheduling algorithm resource allocation only deals with slot assignment and not with wavelength assignment.

Let  $R$  be the  $N \times N$  slot allocation request matrix for unicast traffic, where each entry  $r(s, d)$  with  $s, d \in \mathcal{N}$  represents the number of packets to be sent from source  $s$  to destination  $d$ . Let  $R_w$  be the  $W \times N$  request matrix whose element  $r_w(\lambda, d)$  represents the number of packets that must be delivered to node  $d \in \mathcal{N}$  on wavelength  $\lambda \in \mathcal{W}$ .

In the multicast scenario there are  $M$  multicast groups  $\mathcal{M}_i \subseteq \mathcal{N}$ ,  $i \in \mathcal{M} = \{1, 2, \dots, M\}$ . Note that the matrix  $R$  can be easily extended to the multicast case by using multicast groups instead of single nodes as destinations. We define the  $N \times M$  multicast request matrix as  $\tilde{R}$ , where entry  $\tilde{r}(s, d)$  represents the number of packets to be sent from source  $s \in \mathcal{N}$  to multicast group  $\mathcal{M}_d$ ,  $d \in \mathcal{M}$ .

Several alternatives are available for transmitting multicast packets. We name atomic transmission the possibility of addressing all the multicast packet destinations in the same time slot, thus transmitting only once each packet. Another possibility is multicopy transmission: for each multicast packet addressed to a multicast group comprising  $k$  destinations, the source node creates  $k$  copies; each copy is independently transmitted in different time slots. Finally, in partial multicast transmission, each copy may be partially transmitted to disjoint sub-sets of the multicast group in different time slots.

Let  $\tilde{R}_w$  be the  $W \times M$  multicast request matrix such that element  $\tilde{r}_w(\lambda, d)$  is the number of multicast packets that must be sent from the transmitters on wavelength  $\lambda \in \mathcal{W}$  to multicast group  $\mathcal{M}_d$ ,  $d \in \mathcal{M}$  when using atomic transmission. Finally, let  $R_{wm}$  be the  $W \times N$  matrix whose element  $r_{wm}(\lambda, d)$  is the number of packets that the transmitters on wavelength  $\lambda \in \mathcal{W}$  have to deliver to receiver  $d \in \mathcal{N}$  when using multicopy transmission.

In Sections 2.1 and 2.2 we state the scheduling problem for both unicast and multicast traffic, and derive lower bounds on the frame length required to schedule a given amount of traffic.

### 2.1. Unicast scheduling

The problem of finding an optimum transmission schedule for a given unicast traffic pattern, i.e., the scheduling problem in the off-line context, was already extensively analyzed in [7–13]. It is usually formulated in the following way:

Given a slot allocation request matrix  $R$ , find a time/wavelength assignment that guarantees the delivery of the requested traffic, while minimizing the frame length (i.e., the time necessary to accommodate all transmissions), subject to tuning delay constraints.

Note that frame length minimization implies throughput maximization.

Under unicast traffic a lower bound on the frame length  $F$  can be easily evaluated. Given the request matrix  $R_w$ , the frame length must satisfy the following constraint:

$$F \geq \max \left\{ \max_{d \in \mathcal{N}} \left( \sum_{\lambda \in \mathcal{W}} r_w(\lambda, d) + K_d T \right), \max_{\lambda \in \mathcal{W}} \sum_{d \in \mathcal{N}} r_w(\lambda, d) \right\} \quad (1)$$

where  $K_d$  represents the number of wavelengths on which node  $d$  must tune within the frame in order to receive all packets directed to it. This bound states that the frame length must be large enough to accommodate both the transmissions of all packets on each wavelength and all packets directed to each destination subject to the tuning latency constraint at the receiver. It was proved in [20] that bound (1) can be achieved by the optimal scheduler when tuning times are negligible.

## 2.2. Multicast scheduling

The optimal off-line scheduling problem in presence of multicast traffic can be formulated as follows:

Given a slot allocation request matrix  $\tilde{R}$ , find a time/wavelength assignment that guarantees the delivery of the requested traffic, while minimizing the frame length (i.e., the time necessary to accommodate all transmissions), subject to tuning delay constraints.

To the best of our knowledge, only in [17] the off-line multicast scheduling problem has been addressed and heuristically solved introducing the concept of “virtual receiver”: physical receivers are partitioned in sets called virtual receivers, and the scheduling algorithm schedules virtual receivers requests. In this paper we extend this approach to improve scheduling performance.

Similarly to the unicast case, the lower bound on the frame length  $F$  can be written as a function of  $\tilde{R}_w$  and  $R_{wm}$  as follows:

$$F \geq \max \left\{ \max_{d \in \mathcal{N}} \left( \sum_{\lambda \in \mathcal{W}} r_{wm}(\lambda, d) + K_d T \right), \max_{\lambda \in \mathcal{W}} \sum_{d \in \mathcal{M}} \tilde{r}_w(\lambda, d) \right\}. \quad (2)$$

	1	2		1	2	3
$\lambda_1$	–	{2, 3}	$\lambda_1$	{2, 3}	–	–
$\lambda_2$	{1, 3}	–	$\lambda_2$	–	{1, 3}	–
$\lambda_3$	{2}	{1}	$\lambda_3$	–	–	{1, 2}

Fig. 1. Example of the non-optimality of atomic scheduling.

The first term in (2) represents the number of packets that have to be received by any receiver, including the tuning latency, when using multicopy transmission, whereas the second term is the number of packets that must be delivered on any channel when using atomic transmission. Note that this lower bound is independent of the scheduling algorithm, and can be computed a priori.

Note that the choice of transmitting all packets in an atomic fashion is often clearly non optimal, as already pointed out in [16]. Consider, for example, a network with three nodes and three wavelengths: nodes 1, 2, 3 send respectively 1 packet to multicast groups  $\mathcal{M}_1 = \{2, 3\}$ ,  $\mathcal{M}_2 = \{1, 3\}$  and  $\mathcal{M}_3 = \{1, 2\}$  on wavelengths  $\lambda_1$ ,  $\lambda_2$  and  $\lambda_3$ ; receivers tuning times are assumed negligible. Clearly a frame of length two is sufficient to schedule all the packets if partial transmissions of packets are allowed (see the left hand side of Fig. 1); however, a frame of length three is necessary if only atomic transmissions of packets are allowed (see the right hand side of Fig. 1).

When tuning times are not negligible, it is important to reduce the receiver inefficiency by minimizing the number of times each receiver must tune within a frame to receive all its packets. Most of the heuristic scheduling algorithms defined for unicast traffic thus rely on the minimization of tunings at receivers within a frame, so that each node tunes at most once on each wavelength within a frame. In the heuristic scheduling proposed in this paper, since we are mainly interested in a context in which tuning latencies are not negligible, we enforce that each node tunes at most once on each wavelength within a frame time.

## 3. The multicast scheduling problem

In this section we start with a more formal definition of the multicast scheduling problem in

terms of Integer Linear Programming (ILP). In Section 3.1 we provide a formulation of the scheduling problem in presence of multicast packets as an integer linear program, and show that the number of needed integer variables grows exponentially with the number of network nodes  $N$ . In Section 5.2 we show the optimal schedule for a network having 4 nodes, 2 wavelengths and 6 multicast groups, which is the largest instance of the ILP problem we were able to exactly solve within 24 h of CPU time on a 1 GHz AMD Athlon PC using the CPLEX [21] software.

### 3.1. ILP formulation

Since each multicast packet can be delivered by transmitting several copies on different slot times, each copy being addressed to a sub-set of destinations, for each multicast packet a list of copies must be specified by the scheduler, (i.e., a list of sub-sets of destinations that are reached at the same time). A transmission time is assigned for each copy in the list. More formally, we say that the scheduler associates with each packet a set of elements, each one comprising a sub-set of destinations and a time slot. To reduce the complexity of the scheduler, we suppose that all the packets belonging to the same traffic connection, i.e., coming from the same source and directed to the same group of destinations, originate the same list of copies (i.e., the same partition of the destination set in sub-sets).

We need to define a set of variables to encode the information on the set of copies generated by packets of each flow. For each connection transmitted by source  $i \in \mathcal{N}$  on wavelength  $c_i \in \mathcal{W}$  to the set of destinations  $\mathcal{M}_j$ , whose cardinality is  $|\mathcal{M}_j|$ , and associated with a request  $\tilde{r}(i, j)$ , let us consider the set  $\mathcal{G}_j$  comprising all the possible non empty sub-sets of  $\mathcal{M}_j$ ,  $\mathcal{G}_j = 2^{\mathcal{M}_j} \setminus \emptyset$ . Let  $\mathcal{W}_j = \{i \in \mathcal{N} : c_i = j\}$  be the set of nodes transmitting on wavelength  $j$ ,  $\mathcal{G}_j^k$  be an element of  $\mathcal{G}_j$ ,  $k \in \mathcal{P}_j = \{1, 2, \dots, 2^{|\mathcal{M}_j|} - 1\}$ , and  $\mathcal{B}(j, d) = \{k : d \in \mathcal{G}_j^k\}$  be the set of  $k$  such that  $\mathcal{G}_j^k$  contains physical receiver  $d$ . Note that if  $d \notin \mathcal{M}_j$ ,  $\mathcal{B}(j, d) = \emptyset$ . We define the following binary variables:

$$p_{i,j}^k = \begin{cases} 1, & \text{if for the packets of connection } i, j \\ & \text{a copy is transmitted to } \mathcal{G}_j^k, \\ 0, & \text{otherwise.} \end{cases}$$

The variables  $p_{i,j}^k$  must satisfy the following constraints, which state that each destination belonging to  $\mathcal{M}_j$  must be reached by one and only one copy of the multicast packet:

$$\sum_{k \in \mathcal{B}(j,d)} p_{i,j}^k = 1 \quad \forall i \in \mathcal{N}, j \in \mathcal{M}, d \in \mathcal{M}_j. \quad (3)$$

The binary variables  $t_{i,j}^k(l)$  describe the transmission activity of source  $i$ :

$$t_{i,j}^k(l) = \begin{cases} 1, & \text{if the source } i \text{ transmits a copy} \\ & \text{of a packet to } \mathcal{G}_j^k \text{ in slot } l, \\ 0, & \text{otherwise.} \end{cases}$$

The variables  $t_{i,j}^k(l)$  must satisfy the following constraints, deriving from the fact that no more than one packet can be transmitted in each time slot on each wavelength (5), no receiver can be tuned on more than one wavelength in the same time slot (6), and from the tuning latency constraint (7):

$$t_{i,j}^k(l) \leq p_{i,j}^k \quad \forall i \in \mathcal{N}, j \in \mathcal{M}, k \in \mathcal{P}_j, l, \quad (4)$$

$$\sum_{i \in \mathcal{W}_h} \sum_{j \in \mathcal{M}} \sum_{k \in \mathcal{P}_j} t_{i,j}^k(l) \leq 1 \quad \forall h \in \mathcal{W}, l, \quad (5)$$

$$\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} \sum_{k \in \mathcal{B}(j,d)} t_{i,j}^k(l) \leq 1 \quad \forall d \in \mathcal{N}, l, \quad (6)$$

$$\sum_{i \in \mathcal{W}_b} \sum_{j \in \mathcal{M}} \sum_{k \in \mathcal{B}(j,d)} t_{i,j}^k(l + \tau) + \sum_{i \in \mathcal{W}_a} \sum_{j \in \mathcal{M}} \sum_{k \in \mathcal{B}(j,d)} t_{i,j}^k(l) \leq 1 \\ \forall a, b \in \mathcal{W} : a \neq b, \forall d \in \mathcal{N}, \tau \in \{1, \dots, T\}, l. \quad (7)$$

Since all traffic requests in  $\tilde{\mathcal{R}}$  must be delivered to destinations, the variables  $t_{i,j}^k(l)$  must satisfy the following constraint:

$$\sum_l \sum_{k \in \mathcal{B}(j,d)} t_{i,j}^k(l) = \tilde{r}(i, j) \\ \forall i \in \mathcal{N}, j \in \mathcal{M}, d \in \mathcal{M}_j. \quad (8)$$

Finally, we introduce the binary variables  $e^d(l)$  to identify an upper bound to the frame length. Each variable  $e^d(l)$  is a boolean indicator of the fact that node  $d$  has received all the packets in the current

1	2	3	4	5	6	7	8	9	10	11	12	13	14
{1, 4}	{2}	–	{1, 4}	{1, 4}	{4}	{4}	{4}	{2, 3, 4}	{3}	{3}	{2, 3, 4}	{2, 3, 4}	{3}
2 → 3	1 → 4		2 → 3	2 → 3	2 → 6	2 → 6	1 → 6	1 → 1	1 → 5	2 → 5	1 → 1	1 → 1	1 → 5
{3}	{3}	{3}	{3}	{3}	–	–	–	{1}	–	–	{1}	–	–
4 → 2	4 → 5	4 → 5	4 → 2	4 → 5				4 → 2			4 → 2		
15	16	17	18	19	20	21	22	23	24	25	26	27	
{3}	{1, 3}	{1, 3}	{1, 3}	{3}	{1, 3}	{3}	{1, 3}	{3}	{3}	–	–	–	
2 → 5	2 → 2	2 → 2	2 → 2	1 → 5	2 → 2	2 → 5	2 → 2	1 → 5	2 → 5				
–	–	{4}	{2}	{2}	{2}	{2}	{2}	{4}	{4}	–	–	–	
		3 → 6	4 → 4	3 → 4	3 → 4	3 → 4	4 → 4	3 → 6	3 → 6				

Fig. 2. Optimal scheduling.

frame, and is ready to start the reception in the next frame, having already tuned to the wavelength to be used at the beginning of the next frame:

$$e^d(l) = \begin{cases} 0, & \text{if node } d \text{ is ready to receive in} \\ & \text{the next frame at slot } l, \\ 1, & \text{otherwise.} \end{cases}$$

Variables  $e^d(l)$  permit to obtain a linear expression for the function to be minimized. The variables  $e^d(l)$  must meet the following constraints:

$$e^d(l) \geq e^d(l+1) \quad \forall d \in \mathcal{N}, l, \quad (9)$$

$$e^d(l) \geq t_{i,j}^k(l) \quad \forall i \in \mathcal{N}, j \in \mathcal{M}, d \in \mathcal{M}_j, k \in \mathcal{B}(j, d), l, \quad (10)$$

$$e^d(l+T-\tau+1) \geq t_{i,j}^k(\tau) + t_{a,b}^h(l) - 1$$

if  $\exists i, a \in \mathcal{N} : c_i \neq c_a,$   
 $\exists j, b \in \mathcal{M}, k \in \mathcal{P}_j, h \in \mathcal{P}_b : d \in \mathcal{G}_j^k \cap \mathcal{G}_b^h,$   
 $\forall \tau \in \{1, \dots, T\}.$  (11)

Note that inequalities (9) and (10) force  $e^d(l) = 1$  for all the slots from the beginning of the frame length to the end of the last reception by node  $d$ .

Since a given slot assignment determined by the scheduling process is repeated in consecutive frames until a new traffic matrix  $\bar{R}$  is available, enough time slots must be available at the end of the current frame to allow the receivers to tune to a different wavelength at the beginning of the next frame. Inequality (11) takes care of this by forcing  $e^d(l) = 1$  in the correct time slots depending on when node  $d$  receives the last packet at the end of the current frame, and when it will receive the first packet at the beginning of the next frame. Note

that a number of idle time slots ranging between 0 and  $T$  may be added at the end of the frame (e.g., see the schedule in Fig. 2, where  $T = 3$ ).

With the above definitions, the criterion for the optimization of the scheduling is the minimization of the frame length (i.e., transmission period)  $F$  satisfying:

$$F \geq \sum_l e^d(l) \quad \forall d \in \mathcal{N}. \quad (12)$$

#### 4. Heuristic solutions to the multicast scheduling problem

In this section, we first provide a brief description of the algorithms we propose to solve the multicast scheduling problem. Then we briefly describe the Tabu Search approach used in some of these algorithms. Finally, the scheduling algorithm named Greedy Maximum Weight Matching (GMWM) is presented and its computational complexity is evaluated.

##### 4.1. Heuristic multicast scheduling

The heuristics proposed in this paper are based on an extension of the concept of virtual receiver introduced in [17]. According to the scheme proposed in [17], physical receivers are first partitioned into sets called virtual receivers; all physical receivers belonging to the same virtual receiver tune at the same time on the same wavelength. Since the physical receiver partitioning is the same on all wavelengths, the multicast scheduling

problem is changed into a unicast problem where any unicast scheduling algorithm can be used to determine the slot allocation. We use the GMWM algorithm presented in Section 4.3 instead of that proposed in [17]. We label this solution as Channel Independent Virtual Receivers (CIVR) policy.

In our approach, we relax the constraint of using the same physical receiver partition on all wavelengths, allowing the receivers to be partitioned in different ways on different wavelengths. Let  $V^\lambda \in \mathcal{N}$  be the number of virtual receivers on wavelength  $\lambda$  and let  $\mathcal{V}^\lambda = \{1, 2, \dots, V^\lambda\}$  be the set of the virtual receivers' indices on wavelength  $\lambda$ . Denoting with  $\mathcal{V}_k^\lambda \subseteq \mathcal{N}$  the  $k$ th virtual receiver on wavelength  $\lambda$ , the virtual receivers on wavelength  $\lambda$  satisfy the following constraints:

$$\bigcup_{k \in \mathcal{V}^\lambda} \mathcal{V}_k^\lambda = \mathcal{N},$$

$$\mathcal{V}_i^\lambda \cap \mathcal{V}_j^\lambda = \emptyset \quad \forall i, j \in \mathcal{V}^\lambda : i \neq j$$

and at any given time, only the physical receivers belonging to a given virtual receiver can be tuned on a given wavelength. This approach, named Channel Dependent Virtual Receivers (CDVR) policy, allows a greater flexibility in specifying the virtual receiver partition because each wavelength can be associated with a different number of virtual receivers comprising different physical receivers. The main drawback is that, in general, the intersection between different virtual receivers on different wavelengths is not empty; therefore a more complex scheduling algorithm is needed.

Once the virtual receiver sets on all wavelengths have been defined, it is possible to obtain a lower bound on the scheduling frame length. Recall that physical receivers belonging to the same virtual receiver on a given wavelength  $\lambda$  tune at the same time on wavelength  $\lambda$ . Let  $\bar{R}_w$  be the request set whose element  $\bar{r}_w(\lambda, k)$  represents the number of packets that must be transmitted on wavelength  $\lambda$  to members of virtual receiver  $\mathcal{V}_k^\lambda$ . Note that  $\bar{r}_w(\lambda, k)$  comprises all the unicast and multicast packets transmitted to members of  $\mathcal{V}_k^\lambda$ :

$$\bar{r}_w(\lambda, k) = \sum_{s,d} \tilde{r}(s, d)$$

$$\forall s \in \mathcal{W}_\lambda \text{ and } d \in \mathcal{M} \setminus \mathcal{V}_k^\lambda \cap \mathcal{M}_d \neq \emptyset.$$

Let  $B(\lambda, d) = k$  be the index of the only virtual receiver  $\mathcal{V}_k^\lambda$  on wavelength  $\lambda$  containing physical receiver  $d$ . We can compute a lower bound on  $F$  as

$$F \geq \max \left\{ \max_{d \in \mathcal{N}} \left( \sum_{\lambda \in \mathcal{W}^d} \bar{r}_w(\lambda, B(\lambda, d)) + K_d T \right), \right. \\ \left. \max_{\lambda \in \mathcal{W}} \sum_{k \in \mathcal{V}^\lambda} \bar{r}_w(\lambda, k) \right\} \quad (13)$$

where  $K_d$  is the number of wavelengths on which physical receiver  $d$  must tune in a frame to receive all packets directed to it. This lower bound holds for both CIVR and CDVR policies.

In [17], given a request matrix, a greedy algorithm is used to determine the set of virtual receivers. In the sequel we will use the Tabu Search meta-heuristic (described in Section 4.2) to determine the best configuration of virtual receivers for both CIVR and CDVR policies, given a known request matrix. We will add a T (Tabu) to their acronyms to identify these (significantly more computationally intensive) algorithms.

Tabu Search based algorithms examine a sub-set of virtual receivers configurations. To determine which solution should be preferred among those examined, it is important to evaluate the solutions. This evaluation process can be performed either by scheduling the transmissions according to the GMWM algorithm described in the next section, or by using bound (13). We will add a label S (scheduling) to the acronyms T-CDVR and T-CIVR to identify the algorithms that evaluate the solutions schedule according to the GMWM algorithm, and a label L (lower bound) for the algorithms that are based on the lower bound. Clearly, algorithms that use the lower bound as the evaluation metric are much faster than those that require a complete scheduling to be performed.

#### 4.2. Tabu Search

We provide here a concise description of the principles of the Tabu Search algorithm. We will later provide more details on the specific implementation of this meta-heuristic with reference to the multicast scheduling problem described in the paper.

Tabu Search is based on a partial exploration of the space of solutions to discover a good solution. Exploration starts from an initial solution that is generally obtained applying a greedy algorithm.

For each solution, a class of “neighboring” solutions is defined. The “neighborhood” is defined as the set of solutions that can be obtained from variations of the current solution: each “neighbor” is obtained from the current solution by applying a perturbation, named “move”.

At each iteration of the Tabu Search algorithm, all (or some) solutions in the neighborhood of the current solution are evaluated, and the best solution among them is selected as new current solution.

A “tabu list” is introduced to prevent the algorithm from deterministically cycling among already visited solutions. The tabu list consists of a fixed-size list recording the last moves that have been performed. While a move is stored in the tabu list, it cannot be used to generate a new solution.

After a given number of iterations, the algorithm returns the best visited solution. Note that the Tabu Search algorithm can be seen as an evolution of the classical search algorithm called Steepest Descent [22], that finds a locally optimal solution; however, Tabu Search can accept also solutions worse than the current solution, and thus may avoid to be captured in local minima.

#### 4.2.1. Tabu settings

Our algorithms exploit the Tabu Search meta-heuristic to find a good configuration of virtual receivers. Four fundamental aspects that must be defined in the Tabu Search algorithm are:

- the definition of the space of solutions;
- the choice of an initial solution;
- the definition of the neighborhood;
- the evaluation of the quality of visited solutions.

We describe the Tabu Search parameter setting for the more complex CDVR policy. In this case, since the solution space defined by all the possible virtual receiver partitions on each wavelength would be too large, we limit its size by considering only the configurations that satisfy the two following heuristic criteria:

- each physical receiver that receives only unicast traffic on a wavelength  $\lambda$  is forced to form an individual virtual receiver on that wavelength;
- a virtual receiver  $\mathcal{V}_k^\lambda$  can only be a sub-set of a multicast group.

In this way, we eliminate many configurations that are “a priori” non optimal, strongly reducing the number of iterations required to perform a significant exploration of the solution space.

The virtual receiver configuration resulting from the application of the algorithm proposed in [17] is used as the initial solution of our search.

The neighborhood of a solution is the sub-set of the configurations satisfying the heuristic criteria stated above which can be obtained either by transferring on the same wavelength a physical receiver from a virtual receiver to another, or extracting a physical receiver from an existing virtual receiver to generate a new virtual receiver on the same wavelength.

#### 4.3. The GMWM scheduling algorithm

Given the configuration of virtual receivers on each wavelength, scheduling of transmissions is performed according to the GMWM algorithm, which is very similar to the one proposed in [7]. The algorithm uses some state variables  $t_{\text{next}}^w$ , where  $w$  spans over wavelengths, and  $t_{\text{next}}^d$ , where  $d$  spans over destinations. Variables  $t_{\text{next}}^w$  and  $t_{\text{next}}^d$  contain the information on the time in which wavelengths and destinations become available for subsequent transmissions.

Let us initialize the set  $\bar{R}_w$  with all the requests  $\bar{r}_w(\lambda, k)$  (packets to be transmitted on wavelength  $\lambda$  to virtual receiver  $\mathcal{V}_k^\lambda$ )

*Step 0*  $t_{\text{curr}}$ ,  $t_{\text{next}}^w$ , and  $t_{\text{next}}^d$  are set to 0.

*Step 1* The largest request  $\bar{r}_w(\lambda, k)$  in  $\bar{R}_w$  such that  $t_{\text{next}}^w \leq t_{\text{curr}}$  and  $t_{\text{next}}^d \leq t_{\text{curr}} \forall d \in \mathcal{V}_k^\lambda$  is allocated in the frame starting from  $t_{\text{curr}}$  and is extracted from  $\bar{R}_w$ .  $t_{\text{next}}^w$  and  $t_{\text{next}}^d$  are updated according to the following algorithm:  $t_{\text{next}}^w = t_{\text{curr}} + \bar{r}_w(\lambda, k)$  and, for each  $d \in \mathcal{V}_k^\lambda$ ,  $t_{\text{next}}^d = t_{\text{curr}} + \bar{r}_w(\lambda, k) + T$ .

*Step 2* If no requests are found in STEP 1,  $t_{\text{curr}}$  is updated:  $t_{\text{curr}} = \min_{d,w} \{t_{\text{next}}^d, t_{\text{next}}^w\}$ .

Step 3 If  $\bar{R}_w = \emptyset$  then  $F = \max_{w,d} \{t_{\text{next}}^w, t_{\text{next}}^d\}$  and STOP; otherwise GOTO STEP 1.

#### 4.4. Computational complexity

Let's assess the computational complexity of GMWM. In the initialization phase at most  $O(NW \log(NW))$  operations are necessary to sort the elements in  $\bar{R}_w$ , so as to create an ordered list of requests. At each iteration, at most  $O(NW)$  operations are necessary to execute Step 1, while Step 2 and Step 3 require  $O(1)$  operations. Indeed, in Step 1 all the elements of the list may be visited, and for each element a check on the possibility to schedule the transmission takes place. Note that the complexity involved in the verification phase depends on the size of the associated virtual receiver, and can require up to  $O(N)$  operations, when the associated virtual receiver contains  $O(N)$  physical receivers; however, in this case the number of elements in the list to be visited is smaller than  $NW$ . Thus the total number of operations involved in Step 1 never exceeds  $O(NW)$ . At most  $NW$  iterations are required to complete the algorithm. As a conclusion, the worst case complexity of the GMWM scheduling algorithm is  $O(N^2W^2)$ .

The computational complexity of the Tabu Search based multicast scheduling algorithms can be easily evaluated given that the size of the neighborhood is  $O(N^2W)$ . However, to limit the computational complexity of the algorithm, it is necessary to restrict the exploration of the neighborhood to only  $K$  solutions picked at random at each iteration. Several simulations performed with the aim of tuning the value of  $K$  showed that when  $K$  is too small, the quality of the solution found is unsatisfactory, whereas when  $K$  is too large, the convergence of the algorithm to the best solution is very slow. Thus, we decided to use  $K = 100$ , a choice that provides a good trade-off between runtime complexity and solution optimality.

The computational complexity of each iteration is dominated by the evaluation cost. The evaluation of each neighbor requires  $O(N^2W^2)$  operations if the GMWM scheduling algorithm is run. Only  $O(NW)$  operations are necessary, on the contrary, if the evaluation of solutions is performed on the basis of lower bound (13).

As a conclusion, if the algorithm stops after  $I$  iterations, and given that the cost to obtain the initial solution is negligible,  $O(IKN^2W^2)$  operations are required if the evaluation of each solution is obtained by scheduling transmissions, while only  $O(IKNW)$  operations are required if the evaluation is obtained by bound (13).

## 5. Results

We report performance results for several scheduling algorithms and network configurations. Before showing simulation results, an example of the optimal scheduling obtained by solving the ILP formulation using commercial software is presented. However, due to the complexity of the problem and the huge number of variables, we were able to obtain the optimal solution only in a very small case. Simulation results are later presented to assess the merits of the algorithms on real size networks.

### 5.1. An example of optimal scheduling

Consider a network with  $N = 4$  nodes in which  $W = 2$  channels are available for data transmission. Node  $s_1$  and node  $s_2$  transmit on channel  $\lambda_1$ , while node  $s_3$  and node  $s_4$  transmit on channel  $\lambda_2$ . The receiver tuning latency is assumed to be  $T = 3$  time slots.

Matrix  $\tilde{R}$ , reported in Table 1, contains the transmission requests, expressed in number of slots, associated with flows from source  $s_i$  to multicast group  $\mathcal{M}_j$ , whose composition is reported in Table 2.

Note that, among the considered destination sets, three contain only one node, and thus correspond to unicast flows.

Table 1  
Transmission request matrix  $\tilde{R}$

	$\mathcal{M}_1$	$\mathcal{M}_2$	$\mathcal{M}_3$	$\mathcal{M}_4$	$\mathcal{M}_5$	$\mathcal{M}_6$
$s_1$	3	0	0	1	4	1
$s_2$	0	5	3	0	4	2
$s_3$	0	0	0	3	0	3
$s_4$	0	2	0	2	3	0

Table 2  
Composition of multicast groups

$\mathcal{M}_1 = \{2, 3, 4\}$
$\mathcal{M}_2 = \{1, 3\}$
$\mathcal{M}_3 = \{1, 4\}$
$\mathcal{M}_4 = \{2\}$
$\mathcal{M}_5 = \{3\}$
$\mathcal{M}_6 = \{4\}$

The resulting optimal scheduling frame, obtained by solving the ILP problem described in the previous section with the CPLEX solver [21] is reported in Fig. 2. In each slot of the frame, the set of receivers tuned on each channel is indicated among brackets, along with the multicast flows that are transmitted. Note that in the table, to simplify the notation, both source and destination sets are indicated only with the ordinal indices (for example  $1 \rightarrow 2$  stands for  $s_1 \rightarrow \mathcal{M}_2$ ).

In this example the frame length equals the lower bound, since node  $s_3$  is either receiving a packet or tuning in each slot of the frame. We remark that the three unused slots at the end of the frame are required to allow nodes  $s_3$  and  $s_4$  to tune, so as to be able to receive packets at the beginning of the next frame.

Note that this optimal solution does not imply any partitioning among multicast groups, since the ILP formulation does not force receiver partitioning on the wavelength basis. Nevertheless, the optimal solution requires independence among different wavelength channels (see the splitting of multicast group  $\mathcal{M}_2$  on wavelength  $\lambda_2$ ), thus partially justifying the heuristic approach pursued in the paper; indeed, the wavelength partitioning approach is adopted to simplify the scheduling problem.

## 5.2. Simulation results

Four Tabu Search based algorithms are compared: Tabu Search-CDVR-scheduling (T-CDVR-S), Tabu Search-CDVR-lower bound (T-CDVR-L), Tabu Search-CIVR-scheduling (T-CIVR-S), Tabu Search-CIVR-lower bound (T-CIVR-L). Moreover, we consider:

- the CIVR policy proposed in [17] scheduled with the GMWM scheduling algorithm;

- the minimum admissible frame length (LB) as defined in lower bound (2);
- the multicopy (MC) algorithm, obtained by replacing each multicast request  $\tilde{r}(s, d)$  from source  $s$  to destination set  $\mathcal{M}_d$  with  $|\mathcal{M}_d|$  unicast requests from source  $s$  to each destination  $i \in \mathcal{M}_d$ .

All the Tabu Search based algorithms use as initial solution the solution obtained with the CIVR policy.

Each considered traffic scenario is associated with a request matrix comprising both unicast and multicast traffic. Unicast traffic, described by a source-destination request matrix  $R$ , is always uniformly distributed. The request size associated with each source/destination pair is uniformly distributed between 0 and 16 slots.

For what concerns multicast traffic distribution, we considered two traffic scenarios, leading to six network configurations, three referring to a video-conference traffic pattern, three to a server distribution traffic pattern:

*Video-24-8*: This network configuration comprises  $N = 24$  nodes uniformly distributed on  $W = 8$  wavelengths, so that three nodes transmit on the same wavelength. Six many-to-many multicast traffic connections are superimposed to unicast traffic. For each connection, traffic source/destinations are picked at random among all network nodes. The average multicast group size has been fixed to 10 nodes. Multicast group sizes are Bernoulli distributed. Multicast slot requests associated with sources joining the multicast group are randomly generated according to a uniform distribution with average 32 and standard deviation 8/3.

*Video-24-12*: This network configuration differs from the Video-24-8 configuration only because nodes are uniformly distributed on 12 wavelengths, so that only two nodes share a wavelength in transmission.

*Video-72-24*: This network configuration comprises 72 nodes uniformly distributed on 24 wavelengths. Also in this case, six many-to-many multicast traffic connections are superposed to unicast traffic. The average multicast group size has been fixed to 30 nodes. Multicast slot requests

associated with each source belonging to the multicast group are randomly generated according to a uniform distribution with average 64 and standard deviation 8/3.

*Server-25-9:* In this network configuration, 24 client nodes exchanging unicast traffic are uniformly distributed on 8 wavelengths. The 9th wavelength is reserved to a multicast server. Three one-to-many multicast connections are superimposed to unicast traffic. The server is the source of the three multicast connections. For each connection, traffic destinations are picked at random among all network nodes. The average multicast group size has been fixed to 15. Multicast slot requests associated with each multicast connection are randomly generated according to a uniform distribution with average 64 and standard deviation 8/3.

*Server-25-13:* This network configuration differs from the Server-25-9 only because nodes are uniformly distributed on 12 wavelengths. The 13th wavelength is reserved to a multicast server.

*Server-73-25:* In this network configuration 72 clients nodes exchanging unicast traffic are uniformly distributed on 24 wavelengths. The 25th wavelength is reserved to a multicast server. Six one-to-many multicast connections routed on the server are superposed to unicast traffic. The average multicast group size has been fixed to 30. Multicast slot requests associated to each multicast connection are randomly generated according to a uniform distribution with average 64 and standard deviation 8/3.

We consider as performance metric the frame length, i.e., the minimum number of slots required to schedule all the requests in  $\tilde{R}$ . We tried to compare fairly all the algorithms; for this reason, results reported on the same plot required the same amount of computational power in terms of CPU time. Thus, the scheduling algorithms were run on the same machine (Pentium III PC at 500 MHz running under Linux), stopping the Tabu Search optimization procedure after a fixed amount of CPU time was passed. However, algorithms based on greedy solutions (e.g., CIVR) which are used for performance comparison require negligible CPU times. Note that CPU times must be used as a measure to assess the relative

performance of algorithms when using comparable CPU times, but they should not be taken as an absolute measure of the CPU times that would be required in a real implementation of a scheduler, given that they are run on a general purpose machine.

Before examining performance results, let us focus on Fig. 3, that reports simulation results obtained by running the T-CIVR-L (left plots) and T-CDVR-L (right plots). We plot, for a randomly chosen request matrix  $\tilde{R}$ , the frame length of the best visited solution, estimated by using lower bound (13) for T-CIVR-L (solid lines) and T-CDVR-L (dashed lines), and computed by the GMWM scheduling (dotted lines). Since the conflicts in the scheduling of transmissions directed to virtual receivers on different channels are neglected in the lower bound, several solutions (i.e., configurations of virtual receivers on different channels) that exhibit a small lower bound entail a large scheduling frame. This is due to the fact that lower bound (13) does not permit a good estimation of the required frame length for all the solutions. A stronger degree of correlation between lower

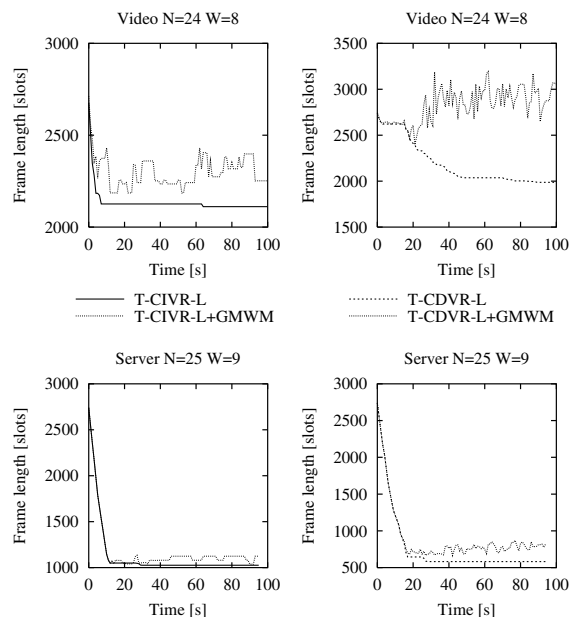


Fig. 3. Frame duration of the best visited solution when running T-CIVR-L (left) and T-CDVR-L (right).

bound and scheduling frame exists for those solutions in which the virtual receiver configuration is independent, or almost independent of the channel, as shown in the top-left plot of Fig. 3, referring to algorithm T-CIVR-L. Finally, note that this phenomenon is less severe when considering the server scenario (bottom plots). This phenomenon may worsen the performance of T-CDVR-L and T-CIVR-L algorithms for increasing CPU time, a clearly undesirable behavior.

To partially cope with this problem, we define the T-CDVR-L\* and T-CIVR-L\* algorithms, which evaluate each solution using lower bound (13); when the best solution (measured using the lower bound) in the neighborhood is found, it is scheduled using the GMWM to determine its quality. Only if the scheduling is satisfactory the solution is kept as the current best solution, since it represents a real improvement over previously examined solutions.

Fig. 4 reports results referring to scenario Video-24-8 (top plots) and Video-24-12 (bottom plots).

Similarly, Fig. 5 reports results referring to scenario Server-25-9 (top plots) and Server-25-13 (bottom plots). All the points were obtained by averaging over 50 different randomly generated request matrices. Tabu Search based algorithms were run for 1 min (left plots) and for 10 min (right plots).

Consider first Fig. 4. All the four Tabu Search algorithms lead to significant performance gain with respect to CIVR and the average frame length is close to the lower bound. Thus, the application of sophisticated and computationally intensive optimization algorithms appears to be rewarded by performance improvements, even for reasonably short CPU times. No major differences in the performance for the four Tabu Search based scheduling algorithms are observable; in general, algorithms that use the scheduling metrics to evaluate the solution perform better than those based on the lower bound. Observing the upper right plot in contrast with the upper left plot, it can be seen that the T-CDVR-L\* algorithm, despite the evaluation of the best solution in the neighborhood using the GMWM algorithm, does not show performance improvements when the CPU

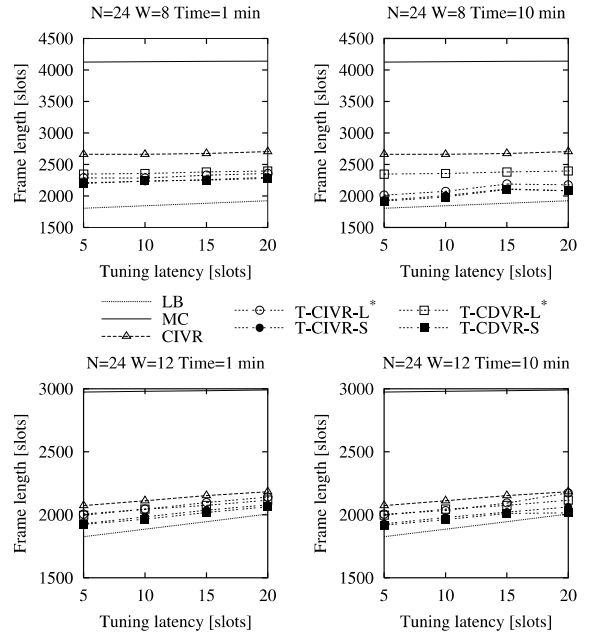


Fig. 4. Average frame length vs latency in the video traffic scenario.

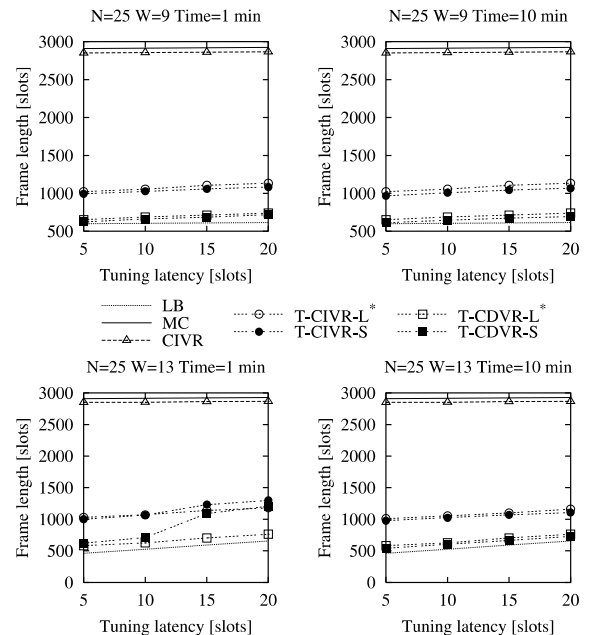


Fig. 5. Average frame length vs latency in the server traffic scenario.

time is increased from 1 to 10 min. In all scenarios, a large performance gain is obtained with respect to the MC algorithm.

Consider now Fig. 5. All the four Tabu Search algorithms lead to large performance gains with respect to CIVR, and the average frame length is close to the lower bound. Moreover, both T-CDVR algorithms perform better than T-CIVR algorithms. It is interesting to observe that, in this scenario, evaluating the solution using a scheduling leads almost always to performance benefits with respect to algorithms using the lower bound metrics. The only exception is for T-CDVR-S when run for 1 min (bottom left plot); the reason is that the CPU time limit is too short to allow the algorithm to determine a good solution. As a matter of fact, algorithms based on scheduling obviously require larger CPU times to evaluate a solution. Clearly, given a limited CPU time, we are able to evaluate a higher number of solutions if using the lower bound metrics instead of scheduling. This means that, if we have very severe time constraints to find a good solution, it may be wiser to choose algorithms based on lower bound metrics; otherwise, scheduling metrics should be obviously preferred.

To evaluate the scalability of algorithms based on scheduling metrics, we plot in Fig. 6 the frame size obtained with the four Tabu Search based algorithms in a large network, i.e., considering the Video-72-24, and Server-73-25 traffic scenarios for one particular request matrix. We plot the best solution found by the algorithms as a function of CPU time, running the algorithms up to 1 h, for  $T = 10$  slots. Interestingly, the algorithms based

on scheduling metrics converge to a good solution in a reasonable time (obviously still larger than the time required by lower bound based algorithms). Note that whereas the T-CDVR algorithms perform much better than T-CIVR in the server scenario, minor differences can be observed when considering the video conference scenario. Since this result is obtained using a single request matrix, we examined the same scenario averaging over 15 different request matrices for a CPU time fixed to 15 min. Results are reported in Table 3, and similar observations can be drawn.

To estimate the speed of convergence of the T-CIVR and T-CDVR algorithms, we ran them on a set of 20 randomly selected instances of the Server-25-13 traffic for  $T = 5, 10, 15, 20$  taking note of the average time required to obtain a solution which is within 2%, 4%, 6% and 8% of the best solution computed by the algorithm in 10 min of CPU time. These results are shown in Fig. 7, the curve labeled “Optimum” gives the average time at which the solution hits the minimum value which is then kept until the end of the run. The figure shows that the T-CDVR algorithm running time is usually smaller than the T-CIVR running time, confirming that the CDVR problem is, in general, easier than the CIVR optimization problem. However, the T-CIVR algorithms obtain a good solution within 2% of the presumed optimum in about 100 s.

The curves plotted in Fig. 8 are obtained in the same way for 10 instances of the Server-73-25 traffic when the reference value for the “Optimum” solution is computed in 60 min of CPU time. Here the time needed by the CIVR algorithm to reach a solution within 2% of the presumed optimum ranges between 600 and 1000 s depending on the

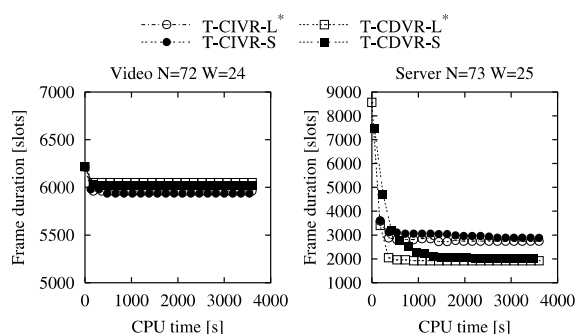


Fig. 6. Frame length of the best solution vs CPU time for a large network.

Table 3  
Average frame length in slot for two traffic scenarios with  $T = 10$

Heuristics	Video-72-24	Server-73-25
LB	5902.0	1765.2
MC	12,871.4	8626.6
CIVR	6855.7	8615.3
T-CIVR-L*	6552.6	2810.3
T-CIVR-S	6515.3	2907.7
T-CDVR-L*	6558.5	1936.2
T-CDVR-S	6570.6	2245.1

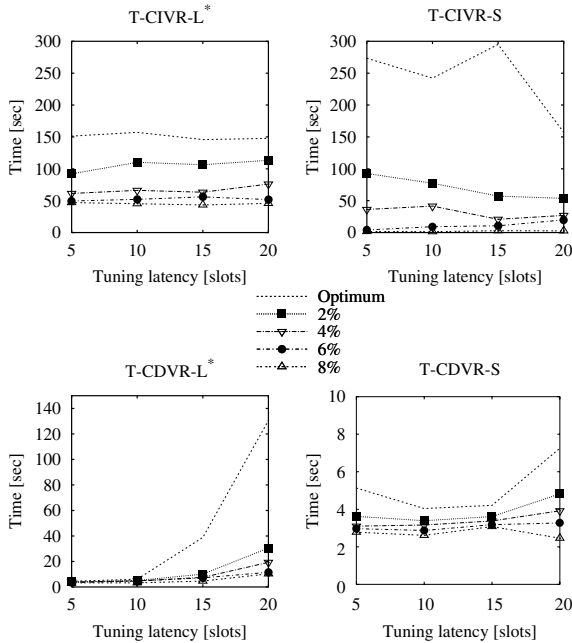


Fig. 7. CPU time vs tuning latency for server  $N = 25$ ,  $W = 13$ .

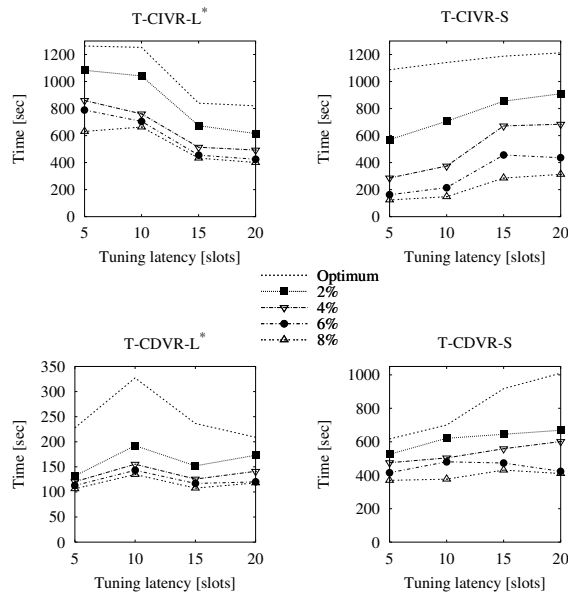


Fig. 8. CPU time vs tuning latency for server  $N = 73$ ,  $W = 25$ .

tuning time, provided that the network and therefore the solution space are larger.

## 6. Conclusions

We have proposed and examined by simulation algorithms that solve the problem of scheduling multicast traffic in TDM/WDM broadcast-and-select networks. We have highlighted the performance benefits that can be obtained by using computationally intensive optimization algorithms based on Tabu Search.

The application of this type of algorithms is rewarded by the significant performance improvements that can be obtained with respect to much simpler greedy algorithms, even by limiting CPU times to reasonably small values. No large differences in the performances for the four Tabu Search based scheduling algorithms are observable.

In general, algorithms that use the scheduling metrics to evaluate the solution perform better than those based on the lower bound, specially when the CPU time constraints are not strict. When CPU constraints are strict, algorithms based on the lower bound are instead a better choice, although in some scenario they exhibit difficulties in converging to the optimal solution.

We have also shown that relinquishing the constraint of channel independency may improve the performance of the scheduling algorithm, specially in the server distribution traffic scenario. All the algorithms have shown good scalability properties for increasing network size.

## References

- [1] A. Bianco, G. Galante, E. Leonardi, F. Neri, A. Nucci, Scheduling algorithms for multicast traffic in TDM/WDM networks with arbitrary tuning latencies, IEEE GLOBECOM 2001, San Antonio, Texas, USA, November 25–29, 2001.
- [2] D. Wonglumson et al., Experimental demonstration of an access point for HORNET; a packet-over-WDM multiple access MAN, IEEE Journal of Lightwave Technology JLT-18 (12) (2000) 1709–1717.
- [3] R. Gaudino, A. Carena, V. Ferrero, A. Pozzi, V. De Feo, P. Gigante, F. Neri, P. Poggiolini, RINGO: a WDM ring optical packet network demonstrator, in: Proc. of ECOC 2001, Amsterdam, Netherlands, September 2001.
- [4] A. Bianco, S. Binetti, P.N. Caponio, A. Hill, E. Leonardi, M. Listanti, F. Neri, R. Sabella, Dimensioning of a single layer optical platform based on the “switchless” concept

- for large scale networks, in: Proc. of IEEE LEOS'99, 12th IEEE Lasers and Electro-Optics Society Meeting, 1999.
- [5] A. Bianco, E. Leonardi, M. Mellia, F. Neri, Network controller design for SONATA a large-scale all-optical passive network, *IEEE Journal on Selected Areas in Communications* 18 (10) (2000) 2017–2028.
- [6] A. Jourdan, D. Chiaroni, E. Dotaro, G.J. Eilenberger, F. Masetti, M. Renaud, The perspective of optical packet switching in IP dominant backbone and metropolitan networks, *IEEE Communications Magazine* 39 (3) (2001) 136–141.
- [7] M. Ajmone Marsan, A. Bianco, E. Leonardi, F. Neri, A. Nucci, Efficient multi-hop scheduling algorithms for all optical WDM broadcast-and-select networks with arbitrary transceivers tuning latencies, in: Proc. of IEEE GLOBECOM'98, Sydney, Australia, November 1998.
- [8] A. Ganz, Y. Gao, A time-wavelength assignment algorithm for a WDM star network, in: Proc. of IEEE INFOCOM'92, Florence, Italy, May 1992.
- [9] M.S. Borella, B. Mukherjee, Efficient scheduling of non-uniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies, *IEEE Journal on Selected Areas in Communications, Special Issue on Multiwavelength Optical Technology and Networks* 14 (5) (1996) 923–934.
- [10] G.P. Pieris, G.H. Sasaki, Scheduling transmissions in WDM broadcast and select networks, *IEEE/ACM Transactions on Networking* 2 (2) (1994) 105–110.
- [11] M. Azizoglu, R.A. Barry, A. Mokhar, Impact of tuning delay on performance of bandwidth-limited optical broadcast networks with uniform traffic, *IEEE Journal on Selected Areas in Communications* 14 (5) (1996) 935–944.
- [12] G.N. Rouskas, V. Sivaraman, Packet scheduling in broadcast WDM networks with arbitrary transceivers tuning latencies, *IEEE/ACM Transactions on Networking* 5 (3) (1997) 359–370.
- [13] I.S. Gopal, C.K. Wong, Minimizing the number of switchings in a SS/TDMA system, *IEEE Transactions on Communications* 33 (6) (1985) 497–501.
- [14] G.N. Rouskas, M.H. Ammar, Multidestination communication over tunable-receiver single-hop WDM networks, *IEEE Journal on Selected Areas in Communications* 15 (3) (1997) 501–511.
- [15] M. Borella, B. Mukherjee, A reservation based multicasting protocol for WDM local lightwave networks, *ICC '95, Toronto, Ont., Canada, June 1995*.
- [16] J.P. Jue, B. Mukherjee, The advantages of partitioning multicast transmissions in a single-hop optimal WDM network, *ICC '97, Montreal, Que., Canada, June 1997*.
- [17] Z. Ortiz, G. Rouskas, H.G. Perros, Maximizing multicast throughput in WDM networks with tuning latencies using the virtual receiver concept, *European Transaction on Communication* 11 (2) (2000); also appeared as a preliminary version, in: Proc. of ACM/SIGCOMM 97.
- [18] H. Lin, C. Wang, Minimizing the number of multicast transmission in single-hop WDM networks, *IEEE ICC'00, New Orleans, LA, USA, July 2000*.
- [19] S. Voss, S. Martello, I.H. Osman, C. Roucairol, *Metaheuristics: Advances and Trends in Local Search Paradigms for Optimization*, Kluwer, Dordrecht, 1999.
- [20] T. Weller, B. Hajek, Scheduling nonuniform traffic in a packet-switching system with small propagation delay, *IEEE/ACM Transactions on Networking* 5 (6) (1997) 813–823.
- [21] <http://www.ilog.com/products/cplex/>.
- [22] G.L. Nemhauser, A.H.G. Rinnoy Kay, M.J. Todd, *Handbooks in Operations Research and Management Science, Vol. I, Optimization*, North-Holland, Amsterdam, 1989.



**Andrea Bianco** is Associate Professor at the Dipartimento di Elettronica di Politecnico di Torino, in Italy. He was born in Torino, Italy, in 1962. He holds a Dr. Ing. degree in Electronics Engineering since 1986 and a Ph.D. in Telecommunications Engineering since 1993, both from Politecnico di Torino. Since 1994 he was an Assistant Professor at Politecnico di Torino, first in the Dipartimento di Sistemi di Produzione, later in the Dipartimento di Elettronica. In 1993 he visited Hewlett-Packard Labs in Palo Alto, CA. In the summer 1998 he visited the Electronics Department at Stanford University, CA. He has co-authored over 100 papers published in international journals and presented in leading international conferences in the area of telecommunication networks. His current research interests are in the fields of protocols for all-optical networks and switch architectures for high-speed networks.



**Giulio Galante** has been with the Electronics Department of Politecnico di Torino, Italy as a Ph.D. student since 2000. He obtained his Dr. Ing. degree in Electronics Engineering from Politecnico di Torino in 1998. During the summer 2000, he was an intern at Lucent Technologies, Bell Labs, Holmdel, NJ. During 2001–2002, he visited the research group of Prof. Nick McKeown at Stanford University, CA. His Ph.D. research is mainly focused on the design of all-optical networks and on the support of multicast in input-queued switches.



**Emilio Leonardi** is Assistant Professor at the Dipartimento di Elettronica di Politecnico di Torino.

He got a Dr. Ing degree in Electronics Engineering in 1991 and a Ph.D. in Telecommunications Engineering in 1995 both from Politecnico di Torino.

In 1995, he was visiting scholar at the Computer Science Department of the University of California, Los Angeles (UCLA). He was visiting researcher at the High Speed Networks Research Department, at Bell Laboratories/Lucent Technologies, Holmdel (NJ) (summer 1999) and at the Electrical Department of the Stanford University (summer 2001), hosted by Prof. Balaji Prabhakar.

He has co-authored over 100 papers published in international journals and presented in leading international conferences, all of them in the area of telecommunication networks.

His research interests are in the field of: performance evaluation of communication networks, all-optical networks, queueing theory, ATM and IP switching.



**Fabio Neri** is full Professor at the Dipartimento di Elettronica of Politecnico di Torino, Turin, Italy. He received his Dr. Ing. and Ph.D. degrees in Electrical Engineering from Politecnico di Torino in 1981 and 1987, respectively. His teaching duties include graduate-level courses on computer communication networks and on the performance evaluation of telecommunication systems. His research interests are in the fields of performance evaluation of communication networks, high-speed and all-optical networks,

packet switching architectures, discrete event simulation, and queueing theory. He leads a research group on

optical networks at Politecnico di Torino. He was general co-chair of the 2001 IEEE Local and Metropolitan Area Networks (IEEE LANMAN) Workshop, and general chair of the 2002 IFIP Working Conference on Optical Network Design and Modelling (ONDM). He has co-authored over 100 papers published in international journals and presented in leading international conferences.



**Antonio Nucci** was born in Lecce, Italy, on 02/02/1974. He graduated from Politecnico di Torino in Electronic Engineering in July 1998. Since November 1999, he has been with the Dipartimento di Elettronica of Politecnico di Torino as a PhD student. He was a Visiting PhD Student starting from September 2000 to December 2000 at the CRT of the Université de Montréal, where he worked with Prof. Teodor Gabriel Crainic and Prof. Brunilde Sansò. Currently he is

working as an employee at ATL Sprint Labs, Burlingame, CA, USA.