

# MULTI-HOP PACKET SCHEDULING IN WDM/TDM BROADCAST-AND-SELECT OPTICAL NETWORKS WITH ARBITRARY TRANSCEIVERS TUNING LATENCIES

M. Ajmone Marsan, A. Bianco, E. Leonardi, F. Neri, A. Nucci  
 {ajmone,bianco,leonardi,neri}@polito.it  
 Dipartimento di Elettronica, Politecnico di Torino, Torino, Italy

*We focus on all-optical slotted WDM/TDM broadcast-and-select networks with  $W$  wavelengths and  $N$  nodes. Nodes are equipped with one tunable transmitter with non-negligible tuning times and one fixed receiver. We first propose a novel scheduling algorithm which exploits the multi-hop transmission technique to shorten the duration of the scheduling period. Then we describe a single-hop scheduling algorithm that performs slightly better than previously proposed scheduling algorithms. A simulation based analysis of the two algorithms shows that they jointly lead to a significant improvement in both throughput and delay with respect to previous single-hop proposals.*

## 1 Introduction

We consider all-optical WDM/TDM broadcast-and-select networks based on a broadcast topology (possibly a star) in which  $W$  wavelengths are available for node-to-node packet communications, and we assume that nodes 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. By tuning the nodes' transceivers, and by dynamically allocating the  $W$  available wavelengths, full connectivity is achieved among end-users. WDM transmission channels are assumed to be slotted and synchronized; each slot can accommodate the transmission of one packet per wavelength.

The time required to tune the node transceivers is not negligible with respect to the packet transmission times; in fact, with the presently available optical components, tuning latency can be quite longer than packet transmission time [1, 2].

We consider protocols based on slowly-varying WDM/TDM transmission schedules that periodically adapt to traffic pattern changes; they provide a good trade-off between simplicity and efficiency. Thus, our scheduling algorithm does not aim at providing a schedule that is modified on the basis of packet-by-packet reservations issued by nodes. Conversely, we assume that node traffic requirements are based on 'sustainable bandwidth' needs, which can be computed with techniques similar to the 'equivalent bandwidth' expressions derived for connection acceptance control in ATM networks [3].

The problem of finding an optimum transmission schedule for a given traffic pattern in presence of significant tuning latencies was already extensively analyzed [4, 5, 6]. It can be formulated in the following way:

GIVEN a requested traffic matrix  $R$ , whose elements  $r_{sd}$  are the numbers of packets that must be transmitted from any node  $s$  to any node  $d$  in a scheduling period, FIND a time/wavelength assignment that guarantees the delivery of the requested traffic, while MINIMIZING the time necessary to accommodate all transmissions, SUBJECT TO technological constraints.

In [4] this scheduling problem was shown to be NP-hard. Several heuristic approaches for the determination of good schedules were proposed in [5, 6].

Most of the scheduling algorithm proposals that appeared in the literature aimed at the minimization of the scheduling period duration assuming that all packets must be transmitted in a single-hop fashion (i.e. packets are directly transferred from their source to their destination with just one transmission). This may not necessarily be the best approach when tuning latencies are long.

In this paper, we present a novel approach for the identification of a good schedule, which yields larger throughput than the schedules resulting from previous proposals. We show that significant throughput gains can be obtained by transmitting selected packet flows in a multi-hop fashion, i.e. by routing them through intermediate nodes. Since we aim at throughput maximization, we do not explicitly optimize delays, and we assume that, in the case of multi-hop transmissions, the successive hops that a packet must traverse are scheduled in successive scheduling periods. Nevertheless, we shall show that the multi-hop schedules allow significant gains in the throughput-delay characteristics.

## 2 The Multi-Hop Scheduling Problem

We consider a network with  $N$  nodes, each node being provided with one tunable transmitter and one fixed receiver. We denote by  $W$  the number of available wavelengths. The transmitters tuning latency, i.e. the time required to tune from a wavelength to a different one, is taken to be  $T$  slots. Thus, two transmissions from the same source at different wavelengths must be separated by at least  $T$  empty slots. Although we consider in this paper the case of transmitter tunability, the proposed approach, can easily be extended to the case of receiver tunability, and to a more general context in which both transmitters and receivers are tunable with significant latencies.

The problem of defining an optimal schedule (i.e. a schedule that maximizes the system throughput) when some traffic flows can be transmitted in a multi-hop fashion, can be formulated as an Integer Linear Programming (ILP) problem, hence shown to be NP-hard [7]. We propose in this paper a heuristic approach to the multi-hop scheduling problem. For simplicity, our heuristic algorithm is presented in the paper in the case  $W = N$ ; extensions to the case  $W < N$  are straightforward.

Recall that the elements  $r_{sd}$  of the requested traffic matrix  $R$  represent the number of packets that must be transmitted from source  $s$  to destination  $d$  in a scheduling period. If we define  $L_i(R)$  as the minimum duration of the activity of node  $i$ , then

$$L_i(R) = \max\left(\sum_j r_{ij} + K_i T, \sum_k r_{ki}\right) \quad (1)$$

where  $K_i$  is the number of destination nodes to which node  $i$  must transmit at least one packet; the length  $L_{sh}$  of the single-hop scheduling period (i.e. the period necessary to accommodate all transmissions in  $R$  in a single-hop fashion) must satisfy:

$$L_{sh} \geq L(R) = \max_i L_i(R) \quad (2)$$

Since we assume that the successive hops of a multi-hop transmission may occur in successive scheduling periods, we can split in two separate steps the problem of defining a packet schedule that minimizes the scheduling period duration, hence maximizes the throughput, while allowing multi-hop transmissions.

The first step is aimed at the design of a logical topology through which packets must be routed from sources to destinations; i.e. at the definition of the set of single-hop transmissions, as well as the routing for multi-hop transmissions.

During the second step, the single-hop scheduling problem is solved starting from a derived requested traffic matrix  $S$ , whose elements  $s_{ij}$  are equal to the numbers of packets that are to be transmitted in a single-hop fashion from node  $i$  to node  $j$ , possibly comprising multi-hop packet flows, according to the solution found in the first step. A novel heuristic single-hop scheduling procedure is proposed, that performs slightly better than previous proposals [5, 6]. It allows the achievement of scheduling period lengths very close to  $L(R)$ , independently from the single-hop requested traffic matrix, and for non-negligible values of the tuning latency.

### The Logical Topology Design Algorithm

The heuristic algorithm for the logical topology design aims at finding a final single-hop matrix  $S$  such that  $L(S)$  is minimized.

The logical multi-hop topology is built by subsequent rerouting of single-hop packet flows in a multi-hop fashion. Let  $S^{(n)}$  represent the single-hop scheduling matrix at step  $n$ , whose elements  $s_{ij}^{(n)}$  are equal to the numbers of packets that are to be transmitted in a single-hop fashion from node  $i$  to node  $j$ , possibly comprising multi-hop packet flows. At step  $n$  of the algorithm, one single-hop packet flow in  $S^{(n)}$ , e.g. the one from  $i$  to  $j$  ( $i \Rightarrow j$ ), is considered, and an

evaluation of the most convenient way to route the flow is attempted. The most convenient route, i.e. the route that leads to a minimization of  $L(S^{(n+1)})$ , is chosen for the transmission of packets belonging to flow  $i \Rightarrow j$ . The routes considered for flow  $i \Rightarrow j$ , are: the single-hop route  $i \rightarrow j$ , and all the two-hop routes through some intermediate node  $k$ .

Define, for each source node  $i$ , the minimum length of the transmission schedule at step  $n$ :

$$L_i^{(n)} = \sum_j s_{ij}^{(n)} + K_i T \quad (3)$$

The logical topology design algorithm comprises the following steps.

1. Initialize  $S^{(0)}$  to  $R$ : all connections are enabled (they take place in single-hop fashion).
2. Among all sources of enabled connections, choose node  $i$  for which  $L_i^{(n)}$  is maximum.
3. Among all destinations of enabled connections whose source node is  $i$ , find the destination node  $j$  for which  $s_{ij}^{(n)}$  is minimum.
4. Find a *pivot* node  $k$  so that  $s_{ik}^{(n)} > 0$  and  $s_{kj}^{(n)} > 0$ , through which the flow  $i \Rightarrow j$  could be routed in a multi-hop fashion; if many such nodes exist, choose the one for which  $L(S^{(n)})_k$  is minimum.
5. If a pivot node has been found, compute the temporary matrix  $S^*$  as follows: first  $S^* = S^{(n)}$ , then  $s_{ij}^* = 0$ ,  $s_{ik}^* = s_{ik}^{(n)} + s_{ij}^{(n)}$ , and  $s_{kj}^* = s_{kj}^{(n)} + s_{ij}^{(n)}$ .
6. If  $L(S^*) < L(S^{(n)})$ , then the rerouting of connection  $(i, j)$  takes place,  $S^{(n+1)} = S^*$ .
7. Connection  $i \rightarrow j$  is disabled. If any enabled connection is left, go to Step 2; otherwise  $S = S^{(n)}$ .

### The Single-Hop Scheduling Algorithm

In the second step, the single-hop scheduling algorithm is executed. The novel algorithm that we propose tries to minimize the number of times a node needs to tune its transmitter in order to send all the packets to be transmitted during a scheduling period. Although this is a reasonable approach for the identification of the optimal schedule, it does not guarantee that the scheduling period length is always minimized.

At the beginning of the algorithm, the first set of simultaneous transmissions in the scheduling period is selected by executing a MWM (Maximum Weight Matching) algorithm (see [8]) on the bipartite  $NW$ -nodes graph that corresponds to the requested traffic matrix  $S$  ( $N$  vertices of the graph correspond to packet sources,  $W$  vertices correspond to packet destinations, and edges are defined by the logical topology). After the completion of some of the transmissions in this initial set and a tuning delay, one or more transmitters may become available for additional packet transmissions on free wavelengths. At this point the MWM algorithm is again executed in

order to match the remaining transmission requests to the idle wavelengths. The algorithm continues until all requested transmissions have been scheduled. Since the algorithm is based on an incremental application of the MWM algorithm, we shall name it IMWM (Incremental MWM).

A more detailed description of the algorithm follows.

For each source-destination pair  $(i, j)$ , define two functions:  $Source(i, j)$  which returns  $i$  and  $Dest(i, j)$  which returns  $j$ . Moreover, let  $\mathcal{A}$  denote the set of assigned connections, i.e. the set of source-destination pairs  $(i, j)$  corresponding to packet flows whose transmission has already been scheduled;  $\mathcal{A}$  is initialized to the empty set  $\emptyset$ . For each source node  $i$ , define the set  $\mathcal{D}(i)$  of destinations corresponding to already scheduled transmissions;  $\mathcal{D}(i)$  is initialized to  $\emptyset$  for each node  $i$ . Let  $t_s$  denote the system time and initialize  $t_s^{(0)} = 0$ . At the algorithm start, all sources and destinations are enabled. The algorithm consists of the following steps.

1. Use a MWM algorithm over the requests issued by enabled source nodes for transmission toward the enabled destinations, in order to find the switching configuration, i.e. the maximal set of concurrently active transmissions; for each pair  $(i, j)$  corresponding to transmissions selected by the MWM algorithm, put  $Dest(i, j)$  in  $\mathcal{D}(Source(i, j))$ , put  $(i, j)$  in  $\mathcal{A}$ , and disable the corresponding source and destination nodes.
2. Update the requested traffic matrix  $S$  by subtracting all the entries corresponding to transmissions on connections that were just added to  $\mathcal{A}$ .
3. For each source node  $i$ , evaluate  $t_i^{(n)}$ , the minimum time at which the node can schedule a transmission toward a destination not belonging to  $\mathcal{D}(i)$ ; let  $t^{(n)}$  be the minimum value of  $t_i^{(n)}$  greater than  $t_s$ . Update  $t_s^{(n+1)} = t^{(n)}$ . Enable the nodes for which  $t_i^{(n)}$  is not greater than  $t_s^{(n+1)}$ .
4. Enable the available destinations in slot  $t_s^{(n+1)}$ , i.e. the destinations that are not receiving any packet in slot  $t_s^{(n+1)}$ .
5. If matrix  $S$  is null, the algorithm terminates. Otherwise, consider  $S_{en}$ , the submatrix of  $S$  referring to enabled source and destination nodes; if  $S_{en}$  is not null, go back to step 1. If  $S_{en}$  is null, go back to step 3.

### 3 Computational Complexity

Let us first examine the computational complexity of the algorithm proposed for the logical topology design.

In the initialization phase,  $L_i(R)$  must be evaluated for each node;  $O(N^2)$  operations are required. Each step of the algorithm basically requires searching for either the minimum or the maximum of  $N$  numbers. The complexity of each step is, as a consequence,  $O(N)$ . The maximum number of steps before termination is  $N^2$ . The complexity of the whole procedure is thus  $O(N^3)$ .

The evaluation of the complexity of the IMWM scheduling algorithm is complex, and only an extremely pessimistic (and thus scarcely interesting) upper bound can be easily derived. This is because the complexity of each step depends upon the size of the bipartite graph on which the MWM search is performed. This in turn depends on the number of elements matched in the previous steps, which is unknown in general. An evaluation of the maximum number of steps required before the algorithm termination is also quite difficult, since this too depends upon the number of matches at each step.

The derivation of a pessimistic upper bound is instead simple if we assume that each match selects just one traffic flow. Since the execution of the MWM algorithm on a  $2N$ -node bipartite graph requires at most  $N^3 \log N$  operations, and the number of steps is at most  $N^2$ , the computational complexity is upper bounded by  $N^5 \log N$ .

Similarly, in the case  $W < N$ , it can be obtained that the computational complexity is upper bounded by  $WN^4 \log N$ .

## 4 Numerical Results

We present numerical results obtained by running the proposed single-hop and multi-hop scheduling algorithms with an approximate implementation of the MWM algorithm. Observe that, although not previously mentioned, we can restrict the algorithm to schedule packets with a limited maximum number of hops.

### 4.1 Example of multi-hop scheduling

Let us begin with an example that shows the operations of the algorithms.

Consider a small all-optical WDM/TDM broadcast-and-select network with  $N = W = 5$ , where nodes and wavelengths are numbered from 1 to 5, each node receives packets at the wavelength labelled with the same number, the tuning time is  $T = 10$  slots, and the requested traffic matrix  $R$  is:

$$R = \begin{pmatrix} 0 & 3 & 4 & 1 & 2 \\ 2 & 0 & 3 & 2 & 3 \\ 1 & 1 & 0 & 5 & 1 \\ 2 & 2 & 1 & 0 & 4 \\ 1 & 2 & 1 & 3 & 0 \end{pmatrix}$$

The requested packet transmissions add up to 44 packets: 3 packets must be transmitted from node 1 to node 2, 4 from node 1 to node 3, and so on.

The scheduling period resulting from the execution of the single-hop scheduling algorithm comprises 50 slots (taking into account the tuning time of the last active source before the next scheduling frame), and is represented in Table 1, where rows correspond to wavelengths, columns correspond to time slots and numbers in the table represent nodes' indices.

Node 2 transmits at wavelength 1 in the first two slots of the scheduling period, then, it tunes to wavelength 3, and, after 10 slot time, it transmits 3 packets; then it tunes to wavelength 5 to transmit 3 more packets, and finally it tunes to wavelength 4 for the transmission of 2 packets. Ten more slots are then necessary to tune again to wavelength 1 and restart the scheduling period.

By running the logical topology design algorithm to construct a 2-hop logical topology (a multi-hop

slot	1	2	3	4	5	...	13	14	15	16	17	...	26	27	28	...	37	38	39	40
wav. 1	2	2	-	-	-	-	-	-	4	4	-	-	5	3	-	-	-	-	-	-
wav. 2	5	5	-	-	-	-	-	-	1	1	1	-	-	4	4	-	-	3	-	-
wav. 3	1	1	1	1	-	-	-	2	2	2	-	-	-	-	-	-	5	-	4	-
wav. 4	3	3	3	3	3	-	5	5	5	-	-	-	-	-	1	-	-	-	2	2
wav. 5	4	4	4	4	-	-	-	-	-	3	-	-	2	2	2	-	-	-	1	1

Table 1: Scheduling obtained on the example by the single-hop IMWM algorithm

slot	1	2	3	4	5	6	7	8	9	10	11	12	...	17	18	19	20	21	22	23	24	25	26	27	
wav. 1	4	4	4	4	4	4	4	3	3	3	3	3	-	-	-	-	-	-	-	-	-	-	-	-	-
wav. 2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	4	4	4	4	4	1	1	1	1	-	-
wav. 3	2	2	2	2	2	2	-	-	-	-	-	-	-	-	-	5	5	5	5	5	5	-	-	-	-
wav. 4	5	5	5	5	5	5	5	5	5	-	-	-	-	-	-	-	-	-	-	3	3	3	3	3	3
wav. 5	1	1	1	1	1	1	1	1	1	1	1	1	-	2	2	2	2	2	2	-	-	-	-	-	-

Table 2: Scheduling obtained on the example by the multi-hop IMWM algorithm

topology with a maximum number of intermediate nodes equal to 1), we obtain the following routing matrix, where each element  $x_{ij}$  represents the intermediate node in the route from source node  $i$  to destination node  $j$ :

$$X = \begin{pmatrix} - & 2 & 5 & 5 & 5 \\ 3 & - & 3 & 5 & 5 \\ 1 & 1 & - & 4 & 1 \\ 1 & 2 & 2 & - & 1 \\ 4 & 4 & 3 & 4 & - \end{pmatrix}$$

From matrix  $X$  we observe that the flow from node 1 to node 3 is routed through node 5, while the flow from node 1 to node 2 is transmitted directly from source to destination, and so on. Entries corresponding to 2-hop flows are emphasized in matrix  $X$ .

The routing matrix  $X$  originates the traffic matrix  $S$ :

$$S = \begin{pmatrix} 0 & 4 & 0 & 0 & 12 \\ 0 & 0 & 6 & 0 & 5 \\ 5 & 0 & 0 & 5 & 0 \\ 7 & 5 & 0 & 0 & 0 \\ 0 & 0 & 5 & 9 & 0 \end{pmatrix}$$

The matrix  $S$  results from the 2-hop routing of a number of packet flows that were previously transmitted in single-hop fashion. For example, the packet to be transmitted from 3 to 2 now goes first from 3 to 1 and then from 1 to 2; this adds 1 to  $r_{31}$  and to  $r_{12}$ , and sets  $r_{32} = 0$ . The number of packets to be transmitted in a single-hop fashion is now equal to 63: 19 packets are now transmitted in 2-hop mode, and their transmissions must be added to the original 44 packets.

Despite the larger number of packets that must be accommodated in one frame, the scheduling period resulting from the execution of the 2-hop scheduling algorithm only comprises 37 slots, and is shown in Table 2.

It can be observed that node 2 transmits at wavelength 3 in the first six slots of the scheduling period, then it tunes to wavelength 5 and transmits 5 packets. The packets that it previously had to transmit to nodes 1 and 4 are now transmitted in 2-hop fashion through nodes 3 and 5. Additionally, node 2 also takes care of the transmission of one more packet that it relays from source node 4 to destination node 3.

## 4.2 Scheduling frame duration

Let us now observe the average performance of the single-hop and multi-hop scheduling algorithms for larger network configurations. We always assume that the number of wavelengths is  $W = 32$ , and we take the number of nodes to be either  $N = 32$  or  $N = 64$ . One hundred different traffic matrices  $R$  are randomly generated, always setting to zero the elements along the main diagonal, and fixing the mean of the distribution from which the other elements are drawn to 10 when  $N = 32$ , and to 5 when  $N = 64$  (so that the average load on each transmission channel is constant). The elements of matrix  $R$  correspond to individual requests from source nodes, and are taken to be geometrically distributed. For  $W = N = 32$ , this is also the distribution of the number of packets transmitted by a node on a given wavelength. With  $N = 64$ , instead, the number of packets that a node has to transmit on a given wavelength corresponds to the superposition of the requests leading to two destination nodes, and is thus distributed according to a two-stage geometric.

The results of the scheduling algorithms are plotted in Fig. 1 as curves of scheduling period duration (averaged over 100 randomly generated matrices  $R$ ) versus the number  $T$  of slots necessary to tune the transmitters. As regards the single-hop scheduling algorithm, we consider both IMWM, newly proposed in this paper, and TAA, originally proposed in [6]. For the multi-hop scheduling, we always consider our heuristic approach for the design of the logical topology, but again we consider either IMWM or TAA for the second step of the procedure.

The curves in Fig. 1 show that the gain obtained with the multi-hop scheduling algorithm can be quite remarkable, specially with long tuning times, both when  $N = W$ , and when  $N > W$ . Instead, the differences between the two considered single-hop scheduling algorithms do not appear to be significant, even if they slightly favor IMWM over TAA.

The multi-hop scheduling algorithm can be restricted to a maximum number of hops to limit the packet delay. In Fig. 2 we show the curves of the average scheduling period duration versus the tuning time in slots, for the case of a maximum number of hops equal to 1 (single-hop), 2, 3, or unlimited (multi-hop). We can observe that improvements are significant from 1 to 2 hops and from 2 to 3 hops,

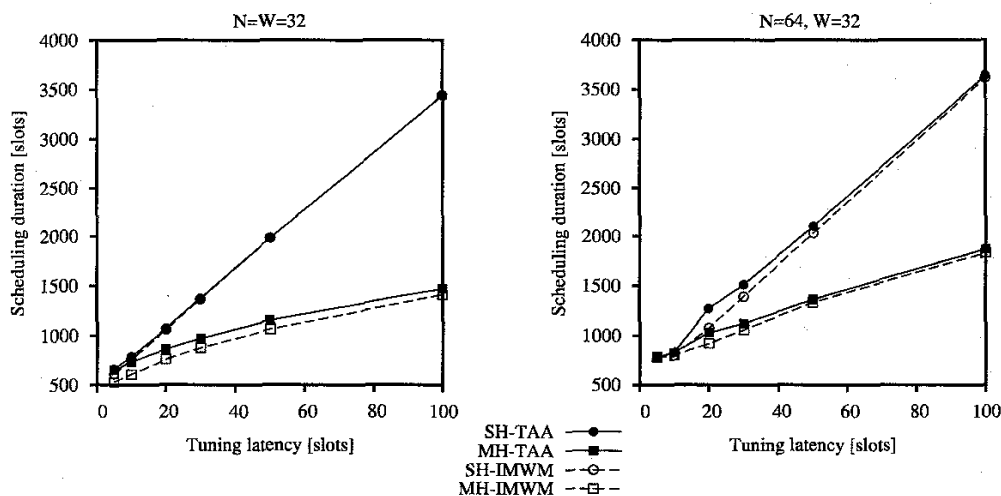


Figure 1: Scheduling period duration as a function of the number of slots necessary to tune node transmitters for the single-hop and multi-hop TAA, and single-hop and multi-hop IMWM algorithms

but there is no evident gain when no restriction on the maximum number of hops is imposed. Actually, the multi-hop case for some tuning delays even produces worse results than the 3-hop case. This is due to the fact that when the number of hops grows, the resulting matrix  $S$  is more difficult to be scheduled in single-hop mode.

#### 4.3 Packet delays

The fact that multi-hop schedules produce shorter scheduling periods with respect to single-hop schedules guarantees that a higher throughput can be obtained. However, it is not clear whether these throughput gains also lead to a reduction of the packet delay, because of the fact that packets must be transmitted several times in successive scheduling periods. Moreover, the fact that gains are large only for long tuning times might suggest, as an alternate approach for the improvement of the system efficiency, the scheduling of a larger quantity of packets (for example scheduling in one scheduling period all the packets corresponding to  $k$  original scheduling periods). While this surely has a negative impact on delay, it might lead to throughput improvements for the single-hop scheduling due to the reduced number of tuning actions.

In order to investigate these effects, we plot in Fig. 3 the curves of the maximum packet delay versus throughput, for the cases  $T = 30$ , and  $T = 50$ . The four curves again refer to the single-hop, 2-hop, 3-hop, and multi-hop schedules. The first point in each curve refers to the average over 30 requested traffic matrices; subsequent points correspond to the averages computed over the same 30 matrices multiplied by a scale factor ( $k = 2, 3, 4, \dots$ ), so as to schedule a larger amount of packets in the same scheduling period. The maximum delay is computed as the scheduling period duration times the maximum number of hops incremented by one.

We can observe that in the considered cases both the single-hop maximum and average packet delays can be greatly reduced with a 2-hop approach over the entire throughput range. The same is not true

for schedules with a larger numbers of hops, that produce quite large gains for low throughput values, but become ineffective at higher throughputs. This means that the gains of the multi-hop scheduling algorithm are maintained, independently of the quantity of traffic to be scheduled, provided that the number of hops is kept low. Multi-hop scheduling, if the number of hops is kept low, leads to significant improvements of the network bandwidth exploitation under urgent strict delay requirements. This result can be explained by observing that the growth in the maximum packet delay is normally proportional to the maximum allowed number of hops, so that imposing a constraint can be quite beneficial.

#### 4.4 Computational costs

Finally, in Fig 4 we plot the maximum CPU times required to run the IMWM algorithms for different values of  $N$  for the case  $W = N$ . We considered a set of 30 randomly generated request matrices. We report for comparison the curves  $\alpha N^5 \log N$  and  $\beta N^3 \log N$ , where  $\alpha$  and  $\beta$  are chosen in order to match the CPU time for  $N = 16$ ; the first curve is related to the pessimistic upper bound computed in Section 3, while the second curve refers to the complexity of the MWM algorithm. It can be seen that the complexity of the logical topology design plus the single-hop scheduling IMWM, measured in terms of CPU times, increases with the number of nodes much less than the pessimistic upper bound computed in this paper, and similarly to the complexity of the MWM algorithm alone.

#### 5 Conclusions

We have considered all-optical WDM/TDM broadcast-and-select networks with  $N$  nodes and  $W$  slotted and synchronized transmission channels. Network nodes have one fixed receiver and one tunable transmitter with non-negligible tuning times.

For these systems, we have proposed to adopt the multi-hop scheduling of WDM/TDM packet transmissions. The multi-hop scheduling problem can be formally defined as an ILP (Integer Linear Program-

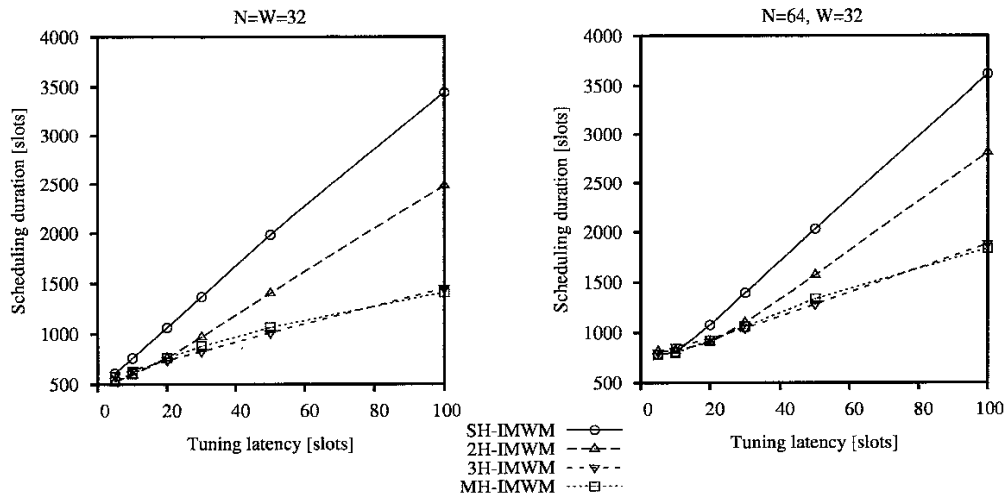


Figure 2: Scheduling period duration as a function of the number of slots necessary to tune node transmitters for the single-hop, 2-hops, 3-hops and multi-hop IMWM algorithms

ming) [7]. In this paper we have described a heuristic approach to the problem, which runs through two steps, called the logical topology design and the single-hop scheduling. An algorithm which provides a sub-optimal solution to the logical topology design problem was proposed. We have further introduced an algorithm for the single-hop scheduling problem which provides minor improvements over previously proposed algorithms.

Our results show that the adoption of multi-hop scheduling algorithms provides important throughput advantages over single-hop scheduling. Limiting the maximum number of intermediate nodes to 1 or 2 leads to a good compromise in terms of throughput vs delay with respect to single-hop and unbounded multi-hop scheduling algorithms.

## References

- [1] A.Sneh, K.M.Johnson, "High-Speed Tunable Liquid Crystal Optical Filter for WDM Systems", IEEE/LEOS'94, Lake Tahoe, NV, USA, July 1994.
- [2] D.A.Smith et al., "Evolution of Acousto-Optic Wavelength Routing Switch". IEEE/OSA Journal of Lightwave Technology, v.14, n.6, pp.1005-1119, June 1996.
- [3] R.Guerin, H.Ahmadi, M.Naghshineh, "Equivalent Capacity and its Application to Bandwidth Allocation in High-Speed Networks", IEEE Journal on Selected Areas of Communications, v.9, pp.968-981, 1991.
- [4] I.S.Gopal, C.K.Wong, "Minimizing the Number of Switchings in a SS/TDMA System", IEEE Transactions on Communications, v.33, n.6, pp.497-501, June 1985.
- [5] A.Ganz, Y.Gao, "A Time-Wavelength Assignment Algorithm For a WDM Star Network", IEEE INFOCOM'92, Florence, Italy, May 1992.
- [6] M.S.Borella, B.Mukherjee, "Efficient Scheduling of Nonuniform Packet Traffic in a WDM/TDM Local Lightwave Network with

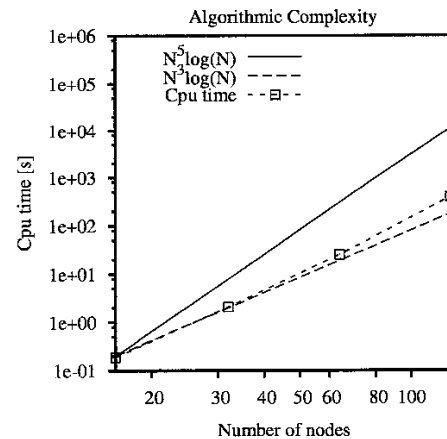


Figure 4: Cpu time for different problem sizes

Arbitrary Transceiver Tuning Latencies", IEEE Journal on Selected Areas of Communications, v.14, n.5, pp.923-934, June 1996.

- [7] M.Ajmone Marsan, A.Bianco, E.Leonardi, A.Nucci, F.Neri, "Multi-hop Scheduling Algorithms for All-Optical WDM/TDM Broadcast and Select Networks with Arbitrary Tranceivers Tuning Latencies", IFIP TC-6 Conference on Optical Network Design and Modeling, Rome, Italy, Feb. 1998.
- [8] N.McKeown, V.Anantharam, J.Walrand, "Achieving 100% Throughput in an Input-Queued Switch", IEEE INFOCOM'96, San Francisco, CA, USA, March 1996.

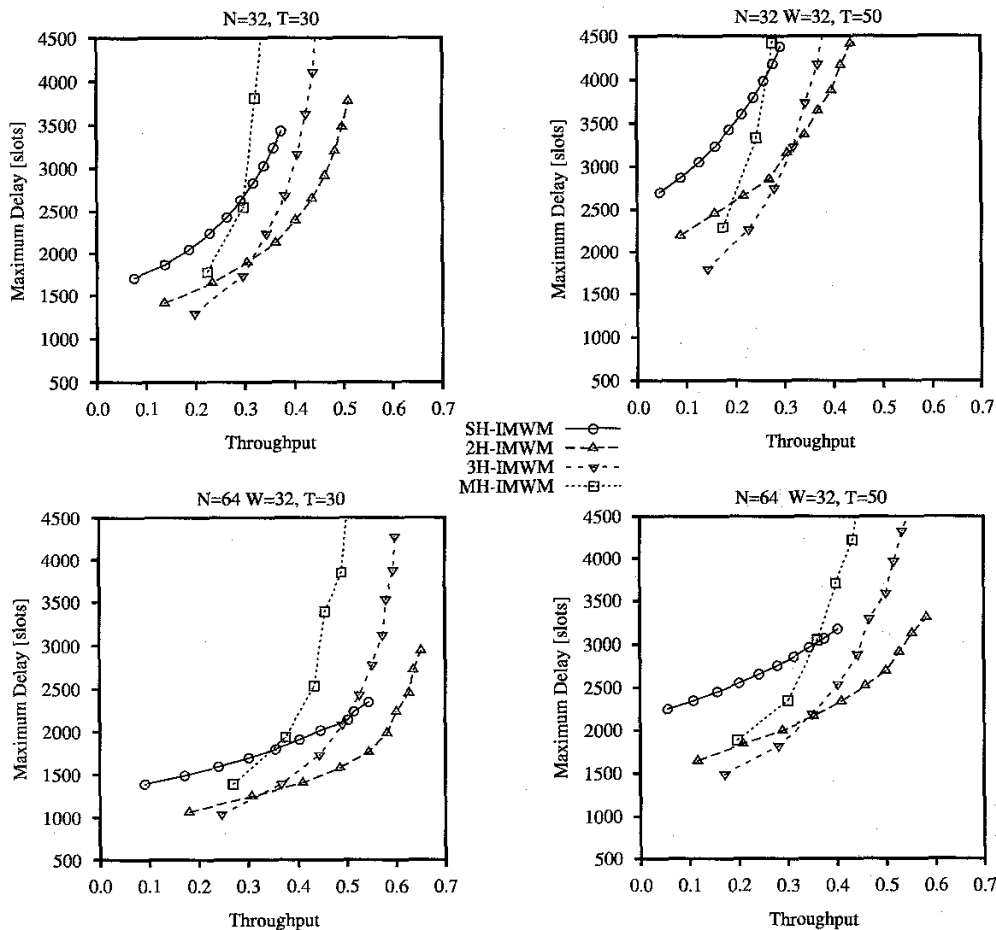


Figure 3: Maximum delay vs throughput for the single-hop, 2-hops, 3-hops and multi-hop IMWM algorithms