Photonic Network Communications (2006) 11:211–227
DOI 10.1007/s11107-005-6024-x

ORIGINAL ARTICLE

CS-POSA: A high performance scheduling algorithm for WDM star networks

P. G. Sarigiannidis · G. I. Papadimitriou · A. S. Pomportsis

Received: 5 August 2004 / Revised: 12 July 2005 / Accepted: 20 July 2005

Abstract In this paper a new packet scheduling algorithm for WDM star networks is introduced. The protocol adopted is pre-transmission coordination-based and packet collisions have been eliminated due to predetermination of the timeslots each node transmits in a demand matrix. The requests of the transmitted packets are predicted through Markov chains in order to reduce the calculation time of the final scheduling matrix. This is accomplished by pipelining the schedule computation. The innovation that this algorithm introduces is to modify the service sequence of the node. The proposed algorithm is studied via extensive simulation results and it is proved that changing the sequence that nodes transmit, from the node with the largest number of requests to the node with the fewest requests, that there is an increase in the throughput of the network, with a minimum (almost zero) cost in mean time delay and in delay variance.

Keywords Optical WDM networks · Star topology · Reservation · Scheduling · Traffic prediction

Introduction

The ever-increasing demand for high speeds in audio, image, and video transmission, within local area network (LAN), metropolitan area network (MAN), and wide area network (WAN) is met by the enormous bandwidth of optic fiber technology. However, optic communication requires subtle operation for the efficient utilization of their possibilities. The crucial problem of this utilization is the co-operation of optic fibers with electronic circuits [1]. The wavelength division multiplexed (WDM) technology, within a single optic fiber [2–4] may result in gigabit-per-second data rates in independent channels that transmit simultaneously data flows to a single or multiple users. Multiplexing as well as demultiplexing of different channels takes place on an optical level, without the inference of electronic circuits and thus increases the capabilities of optic technology in terms of performance, reliability, and control.

Broadcast-and-Select networks [5] comprise a number of nodes and a passive star coupler in order to broadcast from all inputs to all outputs. Every node can select at a given time among the channels available to perform transmission.

In general, if we wish to categorize Broadcast-and-Select architectures based on passive star couplers, we can refer to four possible configurations [6, 7]:

(a) Fixed optical transmitters and fixed optical receivers (FT-FR),
(b) Tunable optical transmitters and tunable optical receivers (TT-TR),
(c) Fixed optical transmitters and tunable optical receivers (FT-TR), and
(d) Tunable optical transmitters and fixed optical receivers (TT-FR).

This paper focuses on the Broadcast-and-Select Star local area network with one tunable transmitter and one fixed receiver (TT-FR) per node (Fig. 1). The network comprises N nodes and W wavelengths.

It can be considered as a general principle of the protocol that the data transmission occurs sequentially in transmission frames, which are divided in timeslots. In each frame, the algorithm examines the transmission requests of the nodes of the network and performs a scheduling process in order to

G. I. Papadimitriou (✉) · P. G. Sarigiannidis · A. S. Pomportsis
Department of Informatics,
Aristotle University,
Box 888, 54124 Thessaloniki, Greece
e-mail: gp@csd.auth.gr

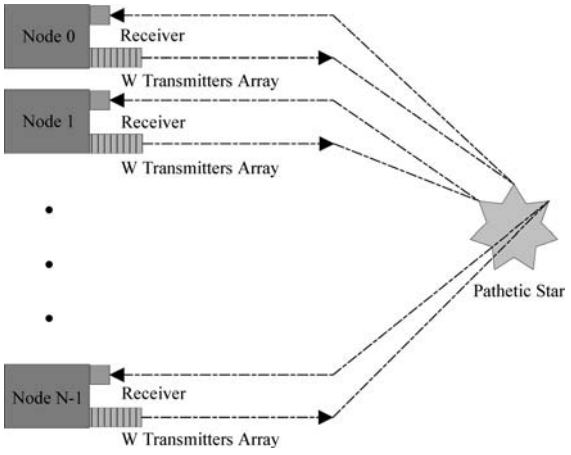

Fig. 1 Broadcast and select star network with tunable transmitter and fixed receiver per node. The network has N nodes and W channels

define the order of the data transmission of each node to the desired transmission channel. There is not any restriction in the number of nodes that can transmit in each frame. Every node can send their requests and have their transmission processed to a timeslot. In addition, every node has access to every transmission channel. However, there are two things that cannot be realized. First, the transmission node itself cannot transmit simultaneously in two channels and second, two or more nodes cannot transmit simultaneously in the same channel. In the FatMAC protocol, which is considered to be the ancestor of this philosophy [8], the transmission of the requests occurs during the reservation phase, while the estimation of the scheduling occurs during the data phase. From then on, there have been improvements of FatMAC [8], as HRP/TSA [9,10], On-line internal-based scheduling (OIS) [11], and Predictive Online Scheduling algorithm (POSA) [12].

The present paper presents an algorithm, check and sort — predictive online scheduling algorithm (CS-POSA), which improves the performance of the above algorithms. Its goal is to eliminate the estimation time of the scheduling that delays the data transmission through its predictions of the requests of the nodes. At the same time, it considers each node individually and serves it according to its workload. Moreover, since the algorithm adopts the method of prediction it must go through some training for a certain amount of time before it starts fully functioning. The most powerful element of the algorithm is that it significantly minimizes the unused timeslots that increase the size of the frame. It consequently increases the performance of the network. This is achieved in a simple way, i.e., by changing the order of service and examination of the nodes while the scheduling matrix is formed. All the above is realized while maintaining the advantage of prediction of the requests of the nodes within a minimum time delay.

The improvement that CS-POSA brings up is presented through a detailed series of figures that represent the results of the simulations both of the channel utilization and the throughput of the entire network. The question whether the algorithm brings extra delay is answered through throughput-delay figures and throughput-delay jitter figure, while the performance of the algorithm is examined in different contexts of network workload through throughput-load figures.

The paper is organized as follows: section 'Network background' analyses the architecture and the structural elements of the network, while section 'OIS and POSA protocols' analyses the two protocols (OIS, POSA) with the previous progress and work to be improved. Section 'CS-POSA' presents the new algorithm CS-POSA, and is followed by the figures and the detailed comparisons between the performance of the two algorithms, POSA and CS-POSA in Section 'Detailed Performance Analysis'. Finally, concluding remarks are given in the sixth section.

Network background

This section describes the structure of the network, providing information on the material used as well as the principles adopted for the operation of the scheduling algorithms.

Network structure

The network comprises N nodes and W channels. Each node disposes an array of tunable transmitters, which provides the transmission of data to the appropriate channels. Moreover, the node has a fixed receiver, which allows receiving data in the particular channel, which is dedicated to each node, known also as home channel. Thus, the network is multicasting and unicasting. The connection of the channels is accomplished through a passive device that allows transparent and immediate transfer of data from the transmitters to the receivers.

It is apparent that the effectiveness of such a TT-FR system strongly depends on the relation between nodes and channels. In general, we can discern four cases: First, the number of nodes is fewer than the channels available ($N < W$). In this case, a number of channels $W - N$ remains idle, since the paths-routes of transmission are ample for the communication of N nodes. This case then, has no practical meaning. Second, the number of nodes and channels is equal ($N = W$). In such a case, the number of paths practically equals the number needed for immediate communication to occur. If we ignore the case where two or more nodes attempt to transmit concurrently and designate the same node as the receiver, then we could argue that this is the ideal case for a direct communication, since each node transmits immediately and without delay the data to the transmission channel, which

is unique for every node. However, this case is rare due to functionality and cost reasons. So, the number of channels is reduced in relation to the nodes of the network, with a consequence to reveal the losses in the utilization of the throughput and in the mean time delay. In the third case then, ($N > W$) a number of nodes, which equals to N/W is shared with a home channel. That is, if we dispose 10 nodes and two channels, it is inferred that five nodes share the same home channel. Finally, it is worth mentioning the forth case where the network disposes a great number of nodes and a very small number of channels ($N \gg W$). It is apparent then, that the application of a very operative algorithm is needed, so as not to burden the network with huge time loss and a small throughput.

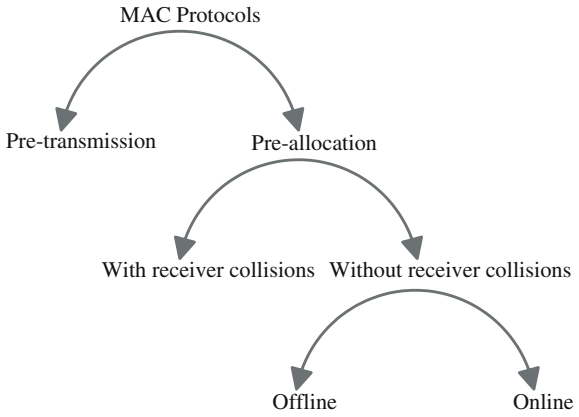

Fig. 2 Categorization of MAC protocols

Categorization of MAC protocols

In a medium access control (MAC) protocol, a series of rules applied to a network are implied so that it can provide communication services between the nodes of the network. Furthermore, due to WDM networks, where the medium of communication is the split into channels, the protocol provides appropriate channel manipulation, so that the delay of information transport between the nodes can be diminished. Apart from that, the channel utilization, which leads to high network throughput, is preserved at high standards.

The objective of MAC protocols is to coordinate the nodes that want to transmit and receive data. This coordination is combined with channel availability. Additionally, node and transmission channels scheduling is resembled to the time unit, so that conclusions can be drawn referring to the overall network performance. Indeed, in order to draw conclusions for the functionality of a MAC protocol, we should refer to the kinds of collisions that appear in WDM Broadcast-and-Select networks under examination. There are two kinds of collisions, under investigation [13]: (a) channel collision and (b) receiver collision. The first kind appears when two or more nodes try to transmit within the same wavelength simultaneously. In such a case, the protocol predicts possible channel collisions; otherwise, it leads to retransmission of packets, since the information is ruined. The second kind of collision appears when two or more nodes try to transmit the data simultaneously to the same node in different wavelengths. Again, in such a case packet retransmission is needed. Undoubtedly, such a case is possible only in architectures with tunable receivers.

A basic distinction among MAC protocols is the existence of a control channel (Fig. 2). In the case where in the network there is at least one channel dedicated to the coordination of channels and their transmission time, then the protocol is based on pre-transmission coordination. In a different case, i.e., if the network does not make use of a separate channel for the node transmission control, the protocol is pre-allocation based. Undoubtedly, at many instances it is observed that protocols do not dispose a separate control channel but exert control through control packets.

The fixed objective of MAC protocols is to eliminate the two kinds of collisions that have been mentioned above. In the pre-transmission case, there are protocols that eliminate the collisions that arise when two or more nodes try to transmit data simultaneously with different wavelengths at the same time towards the same node. Other protocols do not succeed in eliminating the collisions. Representatives of MAC protocols that allow receiver collisions is the family of Aloha [14], slotted Aloha [14], Delayed Slotted Aloha [14], Aloha CSMA [14], Aloha/slotted CSMA [15], DTWDMA [16], Quadro Mechanism [17], and N-DT-WDMA [18].

In all the aforementioned protocols, the initiating mechanism at the packet collision moment, leads to retransmission, a fact that reduces the network performance with apparent negative consequences in bandwidth and delay.

The other category of pre-transmission coordination-based protocols, predicts the possibility of receiver collision and allows one of the participating nodes to transmit at the specific time to the common node-receiver. The other schedule nodes safely transmit their data at different moments. The assurance of successful and safe transmission of data is achieved through a scheduling algorithm, based on the transmitting node's requirements. In other words, the algorithm accepts initially the requirements of all nodes and organizes them in a transmission frame, called traffic demand matrix, $D = [d_{i,j}]$. Time is divided in timeslots. Usually, transmission is organized as frames where each frame is composed of a reservation phase followed by a data phase. The frame then stores for every node the number of timeslots required for transmission to a specific channel. Then the nodes transmit the requested data during the current frame at different moments.

Another important parameter is the size of information that the scheduling algorithm needs in order to function. In the case where the algorithm requires knowledge on the entire

demand matrix D in order to compute the schedule, i.e., requires the requests of every node, then it is characterized as offline. On the other hand, if the algorithm is restrained in partial control information, it can start to operate with the sole knowledge of the requests of the first node N. In this case, the algorithm is characterized as online. It is apparent that, having knowledge on the entire matrix, we get a better and overall view of the requests. Consequently, we can act more efficiently. This means that offline algorithms excel in efficiency, whereas online algorithms excel in matters of time delay between the reservation phase and the transmission phase. This is a fact, because scheduling time is preserved due to the speed of the algorithm, which equals to the entrance of a part of the requests of the node. A further drawback of offline algorithms is the high complexity, which varies from O (M^2C) to O (M^3C^3) [11]. The high levels of complexity do not conform to optic fiber technology and lead to respective long time delays. Besides, high complexity conveys high hardware implementation, like the case of optic technology, where the development of hardware devices is slow and immature. Some examples of offline algorithms are SS/TDMA [19], MULTI-FIT [20], SRA [21], and TAA [21].

OIS and POSA protocols

The next section analyses the two protocols (OIS, POSA) with the previous progress and work.

A brief reference to OIS

The OIS [11] is a typical online algorithm, which exploits the advantages of the algorithms that do not need the whole demand matrix but a part of it. Its online identity is proved by the fact that it starts computing the schedule as soon as the first node sends its set of requests. Thus, the algorithm examines one node after the other with their requests and moves on to construct the scheduling matrix. So, on the one hand it saves time since it starts without delay from the first node, while on the other hand, it is the same algorithm that occurs in every node and in every frame regardless of the workload of each node.

As the network consists of N nodes and W channels, the algorithm functions as following. The moment that a set of requests of the sequential node n is known, then the algorithm examines the availability of the channels for t_1 timeslots transmission that node n requires. In the case that the available channel w is located in the time gap between timeslot t and timeslot $t+(t_1-1)$, then the next step is to examine any potential collisions. In other words, the algorithm checks whether in the timeslots t until $t + (t_1 - 1)$, node n can be scheduled to transmit in another channel $w_1(w_1 \neq w)$. If the registration has been accomplished, then the timeslots t to $t + (t_1 - 1)$ are bound to node n for the wth channel. Thereafter, the lists are renewed and other requests from the remaining $N - n$ nodes are examined. Consequently, the request table (scheduling matrix) of OIS contains for every timeslot the nodes that transmit at that moment and the equivalent transmission channel.

The function of the algorithm is better comprehended, if an example with a given demand matrix is examined. Let us consider the following demand matrix D, 3×3, with three nodes (n_0, n_1, n_2) and three transmission channels (w_0, w_1, w_2).

$$D = \begin{bmatrix} 1 \cdots 2 \cdots 2 \\ 3 \cdots 3 \cdots 1 \\ 5 \cdots 4 \cdots 3 \end{bmatrix}.$$

According to table D, node n_0 requests one timeslot for channel w_0, two timeslots for channel w_1 and two timeslots for channel w_2. Respectively, node n_1 requests three timeslots for channel w_0, three timeslots for channel w_1 and one timeslot for channel w_2. Finally, node n_2 requests five timeslots for channel w_0, four timeslots for channel w_1, and three timeslots for channel w_2. The algorithm does not require to know the rows of all three nodes in order to function, but only the ones of node n_0 [1, 2, 2] (row 1). So, having received the requests of node n_0, the algorithm estimates the time gaps of the channels w_0, w_1, and w_2. Now, one third of the scheduling matrix starts to form (Fig. 3.). It is worth mentioning that the completion of the table is done having also in mind the tuning time of the transmitter that due to simplicity reasons, is considered to be equal to one timeslot. Continuing, the intervals that are disposed to fulfill the requests of node n_1 are estimated, so the second third of the scheduling matrix starts to form (Fig. 4.). Finally, the intervals that are disposed to fulfill the rows of node n_2 are estimated and the scheduling matrix is complete. In Fig. 5, it can be observed that the frame examined lasts for nineteen timeslots. The number of the timeslots in which the channels remain idle controls the performance of the algorithm. It is obvious that channel w_0 transmits data for ten continuous timeslots (or it tunes to a new channel), while it remains idle for nine timeslots idle. Channel w_1 transmits (or tunes to a new channel) for twelve timeslots and the rest remains idle. Finally, channel w_2 transmits only for nine timeslots and the rest remains idle.

Description of POSA

The POSA [12] is a variation of OIS having added the element of the prediction of the requests. The main aim of POSA is to decrease the time of the estimation of the scheduling matrix helped by a hidden Markov chain. With this method, the algorithm succeeds in predicting the requests of the nodes for the subsequent frame based on the requests of the nodes of the previous frames. In this way, time is saved since the

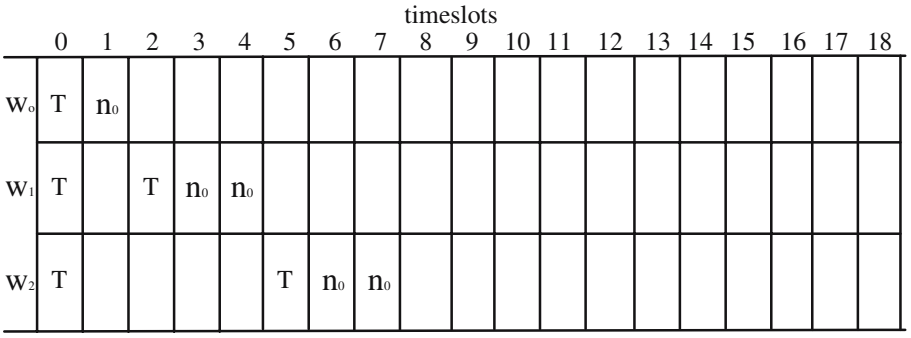

Fig. 3 Scheduling matrix after receiving the requests of node n_0

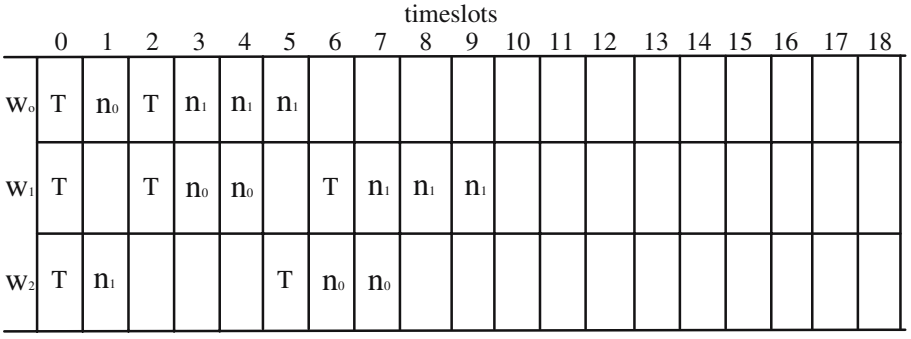

Fig. 4 Scheduling matrix after receiving the requests of node n_1

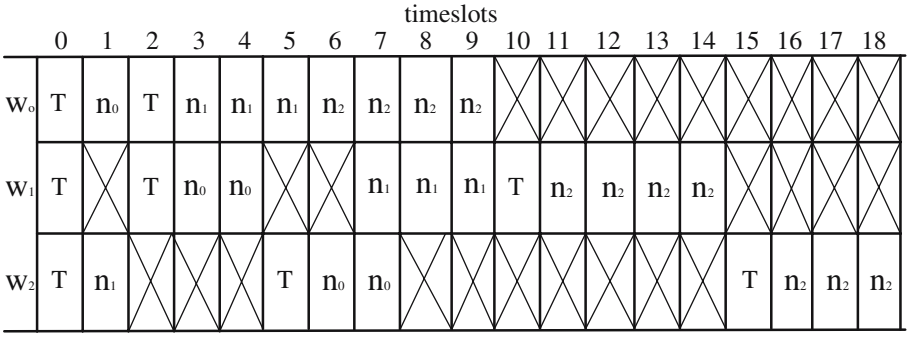

Fig. 5 Final scheduling matrix after receiving all requests

algorithm does not wait for the nodes to send their requests and then to construct the scheduling matrix. Having predicted the requests of the nodes, the scheduling is pipelining at the real time of the transmission of the packets. This parallel elaboration leads to an important—if not complete- decrease of the time of estimation of the scheduling. From the way that the algorithm functions, it is understood that the algorithm works 100% only when the following relation occurs:

$$T\mathrm{p} + T\mathrm{s} \leq T\mathrm{r} + T\mathrm{d},$$

where, Tp is the prediction time, Ts the scheduling operation time, Tr the reservation time, and Td is the actual data transfer time. Apart from this relation that has to be followed in order the algorithm not to decrease its performance, POSA possesses another point that has to be paid special attention to. It is obvious that the most crucial element of the algorithm is its predictor. This predictor after having completed a period of training during which it only learns without predicting, starts predicting based on all the real requests of the nodes. Nevertheless, it is natural for the predictor to make some errors in a small or large percentage. So, it is proved by simulations [12] that the 70% of the predictions have an error rate of less than 20%. This fact allows the use of the POSA method of prediction without dramatic consequences in its performance. The POSA's basic operation occurs in three phases: the learning phase, the switching phase, and the prediction phase. In the first phase, the predictor learns building its history queue, while the reservation, the scheduling, and the transmission of the packets operate just as in OIS. In the next phase, the algorithm moves from learning to prediction, while in the next (ending) phase the algorithm stops constructing the scheduling matrix based on real requests but based on its own predictions for the next frame. Thus, two things occur simultaneously: the prediction of the scheduling matrix for the next frame and the transmission of the packets for the current frame. The algorithm continues the alternation of the situations (prediction-transmission),

while at the same time it builds its history queue so that the predictor is informed of the changes in the traffic of the network.

Before making a brief overview to the operation of the predictor, it is important to mention the three primary assumptions that have to be taken into consideration so that the algorithm is able to predict. First, there must be a predictable underlying pattern to be detected within the $N \times W$ matrix. Second, the number of the requests at each node must be independent, and third, there must be an upper bound K on the requests of every node.

As already mentioned, the predictor is the most important element of the algorithm and assuming that there are N nodes and W various channels, its aim is to build the $N \times W$ traffic matrix, which accept values with a lower bound of 0 and an upper bound of K, i.e., $K + 1$ different states. Fig. 6 shows the form of the traffic matrix after the operation of the predictor. The value $p_{i,j}$ $(0 \leq i \leq N-1, 0 \leq j \leq W-1)$ occurs in the range of values from zero to K, where K is a constant value, so that it is possible to construct a probabilistic-based deterministic predictor. Consequently, the value $p_{0,0}$ is the number of timeslots that the predictor predicted for node n_0, which will be transmitted through channel w_0 in the following frame. Respectively, the value $p_{0,1}$ is the number of timeslots that the predictor predicted for node n_0, to be transmitted through the channel w_1 in the next frame and so on. These values can be from zero to K. Thus, the predictor operates for the rest of the table entries, so it can be argued that there are $N \times W$ different independent predictors that predict the requests of the nodes for the following frame and operate both simultaneously and independently since there is no exchange of information between them.

Each individual predictor $p_{n,w}$ from the $N \times W$ matrix maintains its current state (i.e., the number of timeslots requested by node n in the channel w in the previous frame) in a table with the $K + 1$ different states that may occur at any time. For each of the $K + 1$ different states, i.e., for each one of the entries in the table, the predictor maintains $K + 1$ models that consist of the state transition tables for each possible state and a history queue which records the probability of a change of state of the $p_{i,j}$ from one state to another. Consequently, the predictor $p_{n,w}$ predicts the most probable state from the $K+1$ states based on the $K+1$ models, which it maintains. Of course, in each frame the predictor provides the models with the real states that the nodes send so that it is informed of the changes in the workload in each node. The history queue is maintained for two reasons. First, it is used to provide the predictor a clear picture regarding the current traffic in each $p_{n,w}$ and also to resolve problems in predictive ties. It achieves its first aim by enqueueing at the tail of the queue the most recent state transition and dequeueing from the head of the queue the oldest recording transition. Then, it achieves its second aim by traversing the history queue from the tail of the queue until one of the state transitions within the tie is found. This state is recorded in the traffic matrix for the following frame.

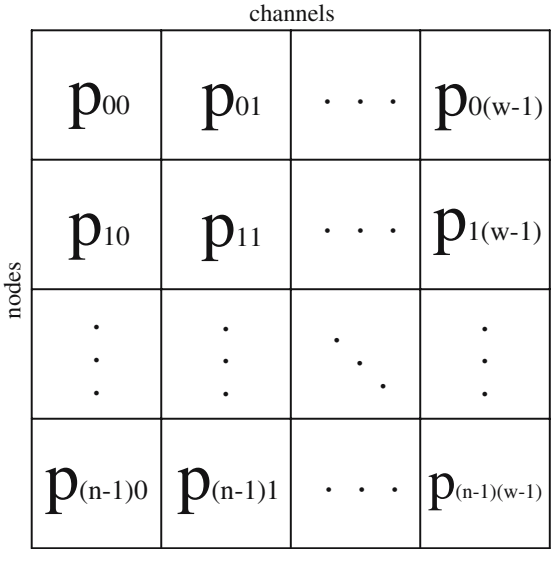

Fig. 6 The traffic matrix after the operation of the predictor

Learning algorithm

The predictor uses two different algorithms, the learning algorithm and the prediction algorithm. During each frame of data, the predictor first runs the learning algorithm and then the prediction algorithm. The first algorithm is responsible for informing and updating the data of the history queue, while the second one is responsible for predicting the demand matrix as accurately as it can.

The learning algorithm is implemented in three steps:

1. At the beginning, the predictor $p_{n,w}[f-1]$, i.e., the predictor of node n which transmits in channel w during frame f of data (Fig. 7), shows the real number of the requests of frame $f-1$, which was requested by node n for channel w. Within the state transition table corresponding to the current state $p_{n,w}[f]$ the algorithm increases the element representing the state in which the predictor actually changed into during the previous frame $(f-1)$.

For example, let us assume that during frame 1980, node 12 for channel 2 requested three timeslots (Fig. 8). Thus, the

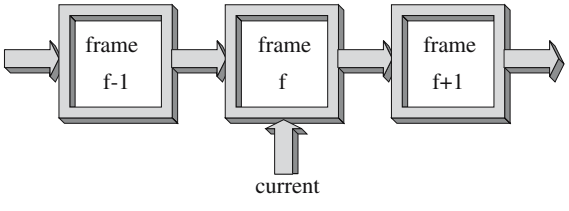

Fig. 7 The sequence of the frames

predictor $p_{12,2}[1980] = 3$. Consequently, it will increase (Fig. 9) for the state $p_{12,2}[1979] = 4$ (i.e., it is equal to the actual number of requested slots for the frame 1979). There is a change in state 3 for the predictor in node 12 and in channel 12.

2. The head of the history queue is dequeued in the position $f - 1 - V$, the oldest transition recorded by the individual predictor $p_{n,w}$ and the element corresponding to the specific state $p_{n,w}$ is decreased.

For example, let us assume that in the position 1980-1000 (considering that the queue is 1000 large), the queue of the state $p_{12,2}[1979] = 4$ shows 0, i.e., it has changed from states 4 to 0, then the probability of change from states 4 to 0 is decreased (Fig. 10).

3. The state of the predictor changes to a state that represents the real number of slots requested by node n in channel w during data frame f.

For example, the predictor $p_{12,2}[1980]$ now shows state 3 which means that the real number of slots requested by node 12 in channel 2 during frame 1980 is three.

Prediction algorithm

The prediction algorithm is implemented in two steps:

(1) Being in data frame f, the algorithm determines the most probable state for the transition of the current state

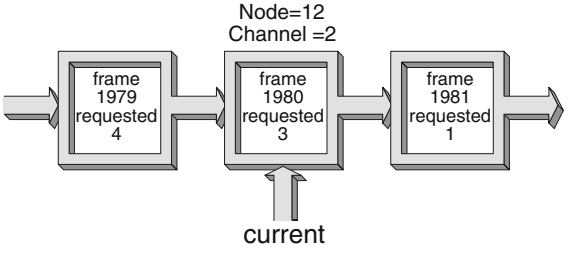

Fig. 8 Operation of the learning algorithm

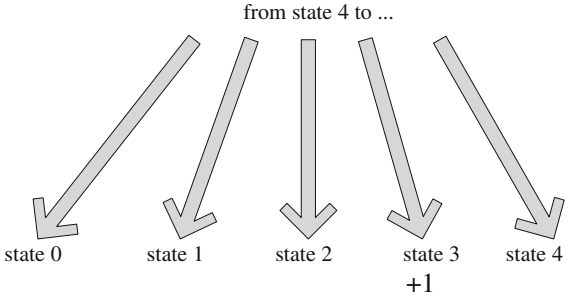

Fig. 9 Initial state transition table for state 4 after update for frame 1979 to state 3

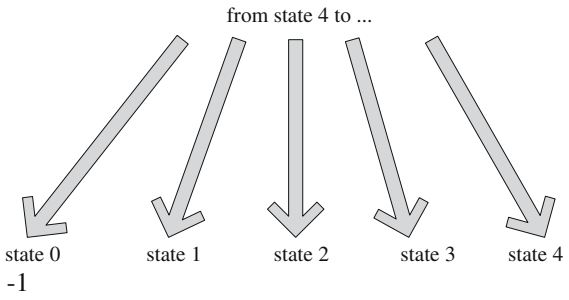

Fig. 10 Initial state transition table for state 4 after update for transition at frame $f - 1 - V$

to be made and gives it as a prediction for the following frame.

For example, the predictor $p_{12,2}[1980]$ equals to three timeslots. So in this phase, the predictor should predict the number of slots that are most probable to be requested by node 12 in channel 2 for data frame 1981. It examines the probability for change of each state and selects the greatest nine (Fig. 11). Thus, for frame 1981 the state 1 will be selected being the right state.

(2) If there is more than one states with the same highest transition count, then the tie is resolved by traversing the history queue from the tail. The first instance of one of the tied transitions encountered within the history queue is the output of the specific predictor.

For example, being in frame 1981, the predictor shows the state 1. Consequently, the algorithm examines, which state is the most probable for the following frame i.e., which state has the highest count of transitions. However, in the figure, three states can be detected (states 1–3) having the same count of transitions and being claimed by the specific predictor (Fig. 12). In a case like this, the algorithm must select the state that is most probable to be chosen for the following transmission frame. So, the algorithm traverses the queue from the tail to find the most recent transition from either

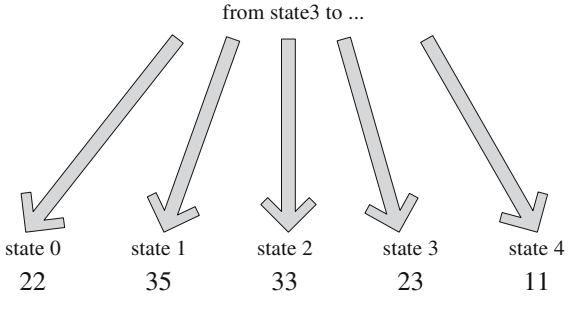

Fig. 11 Initial state transition table for state 3. The predictor outputs state 1, which has the highest state transition count

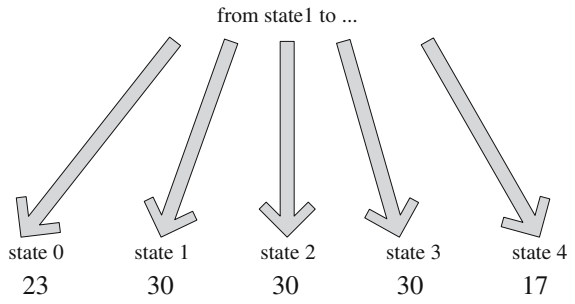

Fig. 12 Initial state transition table for state 1. The predictor must determine to which of these three states (1, 2, or 3) state 1 most recently changed to

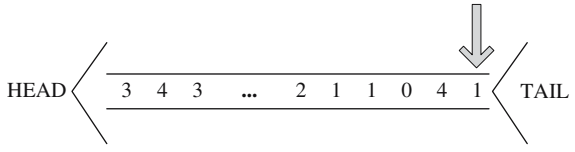

Fig. 13 History queue for state 1. Traversing the queue from the tail the predictor outputs state 1, as the most recent transition

state 1 or 2, or 3. If the tail of the state 1 is the one that is detected in the figure, then state 1 will be chosen (Fig. 13).

CS-POSA

The new proposed algorithm is called CS-POSA. It is based on the two protocols presented in section 'OIS and POSA Protocols', being their continuation and improvement. Its aim is to extend POSA maintaining the pipelining of the schedule computation and the full operation of the predictor. The extension of CS-POSA is based on shifting of the schedule computation of the nodes or in other words, on guiding the order of checking and programming of the nodes. Shifting is based on the workload of each node, which means that the new protocol comprehends better not only the general traffic of the network but also the specific workload in each node. This is accomplished through analytical examination of the requests from each node and then their comparison. So, the grading of the workload of each node is known, as it appears conclusively from all the queues.

Phases of CS-POSA

CS-POSA operates in three phases (Fig. 14):

1. *Learning Phase:* In this phase, CS-POSA learns from the workload of the network how to maintain the history queues. Here, it must be pinpointed that CS-POSA does not examine the workload of each node individually.
2. *Switching Phase:* Here is a change of phases from learning to prediction.
3. *Prediction Phase:* In this phase, CS-POSA predicts the requests of the nodes for the following frame. The innovation that is introduced here is the way of processing the predictions. POSA ignores the variety of the traffic among the nodes building the transmission scheduling matrix starting from the predicted requests of the first node, then the second one and so on until the last one. This is due to the fact that POSA uses OIS to construct the scheduling matrix examining one after the other the requests of the first to the last node. CS-POSA, on the contrary, does not always blindly follow the same service order, i.e., from the first to the last. It examines the summative workload, i.e., the sum of the requests of each node to all destinations and based on it, it processes them in a declining order.

Structural comparison among OIS, POSA and CS-POSA

The structures of the three algorithms are presented comparatively in Fig. 15. It is assumed that real time flows from top to the bottom. Three are main phases, receiving requests from the nodes, estimating the scheduling matrix, and transmission. The contrast between OIS and the other two algorithms is obvious since OIS does not use parallel processing as the other two algorithms do, i.e., the method of pipelining, in order to minimize the computation time of the scheduling. On the one side, POSA carries out simultaneously both the transmission of the data based on the scheduling matrix that the predictor constructed in the previous transmission frame, and the prediction of the scheduling matrix for the following frame. At the same time, there is operation of the learning algorithm that receives the real requests of the nodes and renews the history queue of each predictor. After completing the phase of prediction, POSA constructs the scheduling matrix for the following frame. On the other side, CS-POSA follows the same order and parallelism with POSA but before it constructs the final scheduling matrix it transfers the processing order of the nodes starting from the one with the highest count of requests and finishing to the one with the smallest count. In this way it maintains the parallel processing of the computing schedule phase with the transmission phase, i.e., the real transmission of the data to the nodes.

Predictor of CS-POSA

Assuming that there are N nodes and W channels, each node possesses a set of W queues and each queue can store 0 to K actual requested numbers of slots. The aim of the predictor is to construct the $N \times W$ demand matrix D with N rows

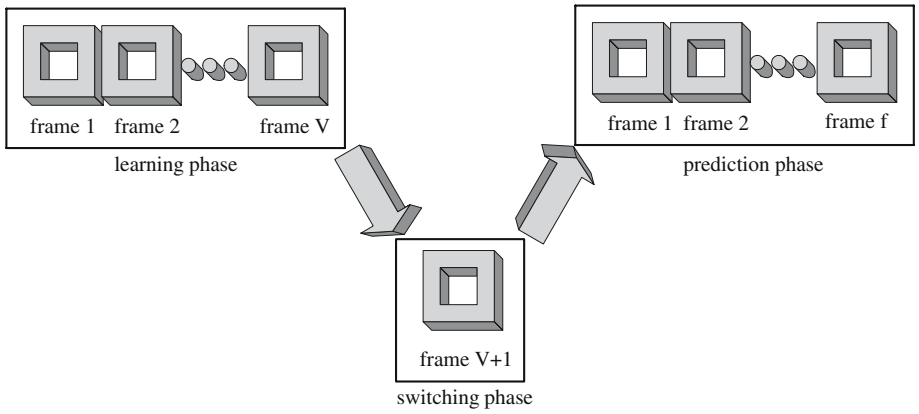

Fig. 14 Phases of CS-POSA

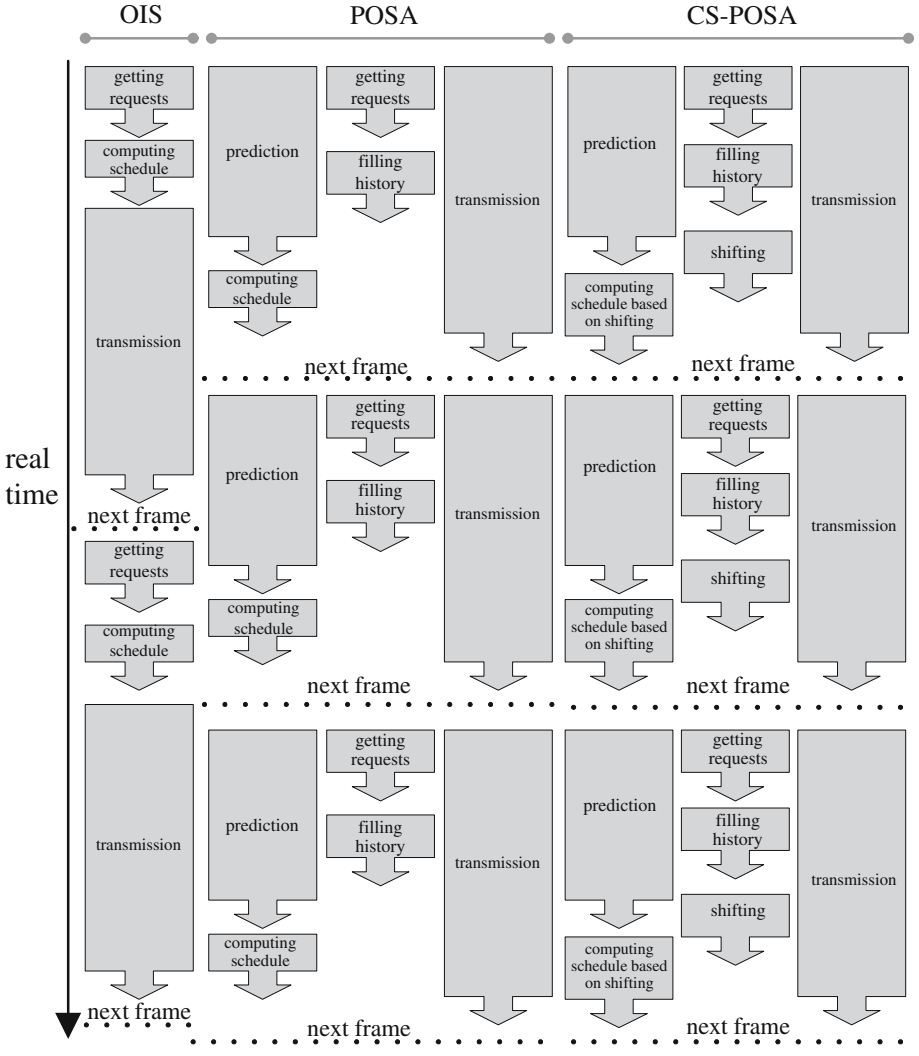

Fig. 15 Structural comparison among OIS, POSA, and CS-POSA

(nodes) and W columns (channels). Each value in the table can vary between 0 and K. Thus, the total predictor looks like $N \times W$ separate and independent predictors whose aim is to predict accurately the expected next number of slots for node n, which transmits in channel w.

The predictor needs the following environment in order to operate:

1. Probability of prediction through a traffic model.
2. Independent and uninfluenced operation of each of the different $N \times W$ predictors.
3. The demand matrix which will be constructed cannot have a value higher than K, where K is a known constant value.

There are four goals that are set for CS-POSA in order to operate effectively. These are:

1. Accuracy of prediction: A high level of accuracy is desired in the predictions. In POSA and CS-POSA, this is achieved through empirical results [12].
2. Real time learning: The predictor should be capable of responding dynamically to changes in network activity. Moreover, CS-POSA comprehends the network activity even better since it receives the requests of each node separately.
3. Asymptotic time complexity: The CS-POSA algorithm must not have a high level of complexity. POSA has an asymptotic time complexity less than O (MC^2K). The same goal is achieved by CS-POSA, too.
4. Scalability: In order to be useful, the algorithm must be scalable with values as N nodes, W channels, an upper bound on K, and the size of the table V. This goal is also achieved both by POSA and by CS-POSA.

Scheduling shifting

In order to understand better the need for studying and co-estimating the individual workload in each node separately, a specific example is examined.
The following traffic matrix has been constructed by nine individual predictors:

$$D = \begin{bmatrix} 1\cdots2\cdots2 \\ 3\cdots3\cdots1 \\ 5\cdots4\cdots3 \end{bmatrix}.$$

It is clear that the predictor $p_{0,0}$ predicted one timeslot for node n_0 with channel w_0, the predictor $p_{0,1}$ predicted two timeslots for node n_0 with channel w_1, and so on. Thus, totally there are:

$p_{0,0} = 1$ node n_0, channel w_0,
$p_{0,1} = 2$ node n_0, channel w_1,
$p_{0,2} = 2$ node n_0, channel w_2,
$p_{1,0} = 3$ node n_1, channel w_0,
$p_{1,1} = 3$ node n_1, channel w_1,
$p_{1,2} = 1$ node n_1, channel w_2,
$p_{2,0} = 5$ node n_2, channel w_0,
$p_{2,1} = 4$ node n_2, channel w_1,
$p_{2,2} = 3$ node n_2, channel w_2.

The POSA operating like OIS, will construct the following schedule matrix (Fig. 16):
As can be seen in the Fig. 16, it is assumed that one timeslot is given to tuning latency time. Before CS-POSA constructs the schedule matrix, it takes the two following steps:

Step 1. Add each row of the traffic matrix D in a new table S that will register the total amount of requests by each node:

$$D = \begin{bmatrix} 1\cdots2\cdots2 \\ 3\cdots3\cdots1 \\ 5\cdots4\cdots3 \end{bmatrix}, \qquad S = \begin{bmatrix} 5 \\ 7 \\ 12 \end{bmatrix}.$$

So, Table S consists of the total amount of the requests of the three nodes for the three transmission channels. Table S is a mirror of the activity that each node has.
Step 2. Grade table S in a declining order. In case those two nodes are found with the same total number of requests, then the selection is random. In this way, vector S changes in the ordered vector S′:

$$S' = \begin{bmatrix} 12 \\ 7 \\ 5 \end{bmatrix}.$$

That denotes that the requests of node n_2 will be first examined, then those of node n_1 and finally those of node n_0. For the same demand matrix, CS-POSA will construct the following scheduling matrix (Fig. 17):
It is clear from both two Figs. 16 and 17 that the scheduling matrix of POSA spends a total of 19 slots from which 26 out of 57, i.e., a percentage of 43% wasted. On the other side, the scheduling of matrix of CS-POSA spends a total of 15 timeslots from which 14 out of 42, i.e., a percentage of 33.33% is wasted. Of course, the case that has been examined is quite specialized. In section 'Detailed Performance Analysis', the two algorithms are compared and contrasted giving results for the utilization of the channels, the throughput of the network, the connection between load and throughput, load and delay jitter, throughput and delay, and throughput and delay jitter. The two algorithms have been measured during 10,000 transmission frames from which the first 1000 belong in the phase of learning.

Algorithm complexity

A substantial element in the function of the algorithm is time complexity. Such a factor is examined since the algorithm has to do with the optical network where the speed is at the maximum. So, the algorithm must keep up with this speed and at the same time it must allow the network some changes in terms of nodes and channels without dramatic consequences in its performance. Moreover, the time complexity of the algorithm must not influence the function of the predictor so that the predictor predicts correctly each change of the nodes and the channels of the network.

It is significant that in POSA, time complexity of the overall predictor is given by:

$$O(K + 1 + V)(NW).$$

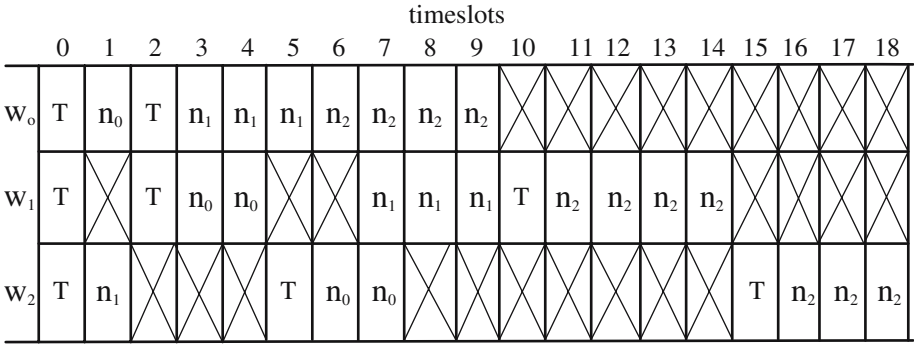

Fig. 16 Scheduling matrix, constructed by POSA

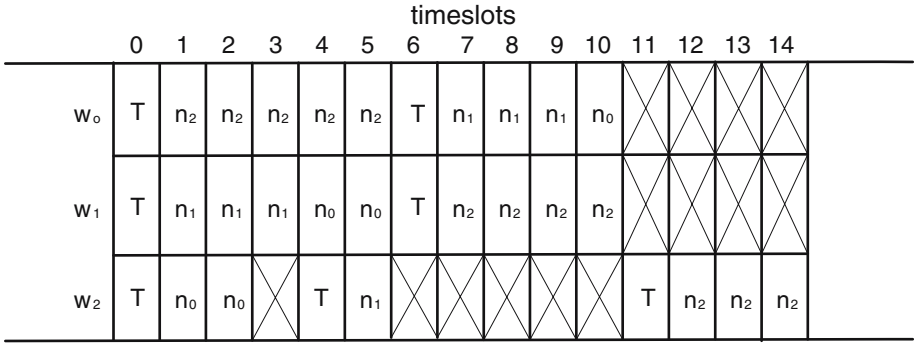

Fig. 17 Scheduling matrix, constructed by CS-POSA

Using P various processors that process the algorithm at the same time, it comes up that, if the factor P is considered to be at the level of NW (i.e., $P = NW/p$), where p is a constant, then:

$$O\left(\frac{(K+1+V)(NW)}{P}\right).$$

The CS-POSA on the other side, maintains each individual predictor at the same value of time complexity:

$$O(K+1+V).$$

In the case that there are a lot of processors so that the predictors can function simultaneously, the overall time complexity is equal to:

$$O(K+1+V).$$

Thus, CS-POSA does not bring any extra complexity in the function of the predictor maintaining at the same time scalability in the system. However, the algorithm also performs a shifting of the load of each node, which, of course, has nothing to do with the function of the predictor. The sorting that is introduced by the algorithm occurs outside the context of the prediction and does not influence at all. So, considering that the shifting has the following complexity:

$$O(N \log N)$$

and co-estimating the fact that the algorithm works with P different processors at the level of NW, then the extra complexity of the CS-POSA is:

$$O\left(\frac{\log N}{W}\right).$$

Of course, the focus is on the local networks where the value of N is at low levels. Anyway, it must be stressed that the extra complexity of the algorithm is minimal and neither delays nor influences the function of the algorithm, since there is the probability of each individual predictor to predict both the workload and the position of the node on table $\grave{S}$ without any classification, maintaining the complexity at the same levels with that of POSA.

Detailed performance analysis

This section presents the performance analysis results. Two algorithms, POSA and CS-POSA, have been studied and analyzed in the context of utilization and throughput, under uniform traffic. Also, the behavior of the two algorithms is presented when the workload of the network is increased in the context throughput-delay, throughput-delay jitter, throughput-load, and delay jitter-load. In the results of the simulation, it is assumed that N is the number of nodes; W is the number of the channels, and K is the maximum value over all entries in the traffic matrix. Also, it should be mentioned that the tuning latency time is considered to be equal to zero timeslots for simplicity reasons.

The simulation took place in a *C* environment. Its duration was 10,000 frames from which the 1000 belong to the learning phase of the algorithms. A random number generator

was used to provide values to the traffic matrix. The values range between 0 and K and in order the goal of scalability to be achieved, the value of K is not constant in the following experiments but each time it is equal to:

$$K = \left\lfloor \frac{NW}{5} \right\rfloor.$$

Measures and measurements that have been studied

In the analysis of the two algorithms, common measures and measurements have been used and are presented below:

(A) Schedule length is symbolized by L and denotes the number of slots in the data phase as determined by the schedule algorithm.
(B) Total slots requested by all nodes are symbolized by R and denotes the total number of timeslots that were requested by all the nodes of the network.
(C) Schedule or channel utilization is symbolized by U and denotes the number of slots actually utilized for packet transmission in a scheduling matrix. Scheduling utilization is defined as:

$$U = \frac{\text{totalslots}}{\text{actualslots} * \text{channels}} \quad \text{or} \quad U = \frac{R}{LW}.$$

(D) Throughput is symbolized by Γ and denotes the average number of bits transmitted per transmission frame per channel. It is measured in Megabit per second. So:

$$\Gamma = \frac{lR}{W(C + Ll/S)}$$

l denotes the packet length in bits, C the computation time in microseconds and S the transmission rate in Mbps. Since the two algorithms, which are examined, do not waste computation delay due to pipelining throughput, the relation finally becomes:

$$\Gamma = \frac{R}{LW} S \quad \text{or} \quad \Gamma = US.$$

(E) Delay is symbolized by D and denotes the mean time delay of the transmitted data in timeslots. It equals to the number of timeslots that pass from the moment that a packet with data is produced in the tails until the moment it is transmitted. If for example, a packet with data has been produced at the time moment t_1 and in the scheduling matrix it has been set to be transmitted at the time moment t_2, where $t_2 - t_1 = t$ timeslots, then $D = t$.
(F) Delay jitter is symbolizes by σ and denotes the variation of the delays with which packets traveling on a network connection reach their destination [22].

Scheduling utilization results

The results from the comparison between the two algorithms are shown in the Fig. 18. It is clear that CS-POSA is obviously improved from POSA. This improvement lasts in all numbers of nodes from 6 to 60 that were simulated both for 8 and 12 channels. The difference between the two algorithms is not greater for eight channels. The biggest difference for eight channels appears to be on the level of 3.7% for 12 nodes, while the smallest reaches 0.5% for 60 nodes. The biggest difference for 12 channels appears to be on the level of 4.45% for 24 nodes, while the smallest reaches 0.9% for 60 channels. The most important conclusion from the comparison between the two algorithms when measuring the schedule utilization is that CS-POSA remains constantly better than POSA for each number of nodes, either for 8 or for 12 channels.

Throughout results

The results from the comparison between the two algorithms on the issue of throughput, are presented in the two following figures for two different speed lines, i.e., for 1.2 (Fig. 19) and 2.4 Gbps (Fig. 20). For 1.2 Gbps the maximum throughput that POSA provides is 11.5 Gbps in 12 channels, while CS-POSA provides 12 Gbps in 12 channels. For 2.4 Gbps the maximum throughput that POSA provides is 23, while CS-POSA provides 23.6 Gbps.

The greatest difference that the two algorithms present is in 12 channels and reaches 1.5 Gbps. It is worth mentioning that again CS-POSA is constantly better that POSA both for 1.2 Gbps and for 2.4 Gbps, regardless of the number of the nodes in the network.

Throughput vs. delay results

The results from the comparison between the two algorithms are presented in the two figures that follow (Figs. 21 and 22). The behavior of the two algorithms is presented, i.e., the relation throughput-delay, altering the values of the workload of the network i.e., of K. The values of K that are tested are: 2, 4, 6, 8, 10, 20, 30, 40, 50, 100, and 200. The number of the nodes is 24, while the available channels are in the first case 8 and in the second case 12. The line speed has been set in 2.4 Gbps.

In the first graph (Fig. 21), it is obvious that there is a constant difference between the algorithms in the context of throughput as the time delay is increased. In other words, it can be observed that for each value of K, both algorithms have almost the same time delay, while CS-POSA is seen improved in the context of throughput. It is typical behavior that when the two algorithms show a time delay of almost 100 timeslots (POSA 98.83 CS-POSA 100.70), their equivalent difference in throughput is improved for the CS-POSA, which reaches 500 Mbps.

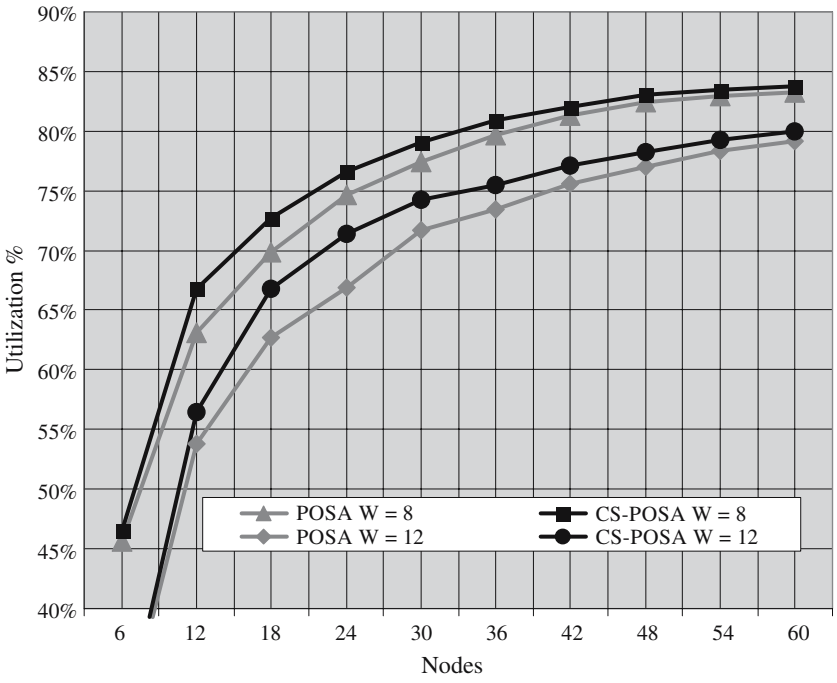

Fig. 18 Channel utilization

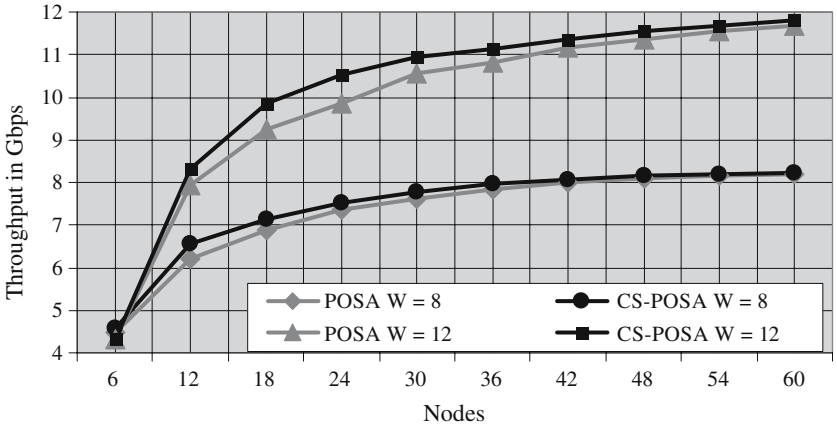

Fig. 19 Throughput (1.2 Gbps)

In the second graph (Fig. 22), the two algorithms are compared in the same K values, with 24 nodes but with 12 channels. The results do not differ greatly from the first graph, since for each value of the workload, CS-POSA precedes POSA without a significant time delay. It is typical that their greatest difference in mean time delay is in eight timeslots (425 for POSA and 433 for CS-POSA) when the value of K is 200, i.e., every position of the traffic matrix can receive values of 0 to 200 timeslots. Of course, with a difference of eight timeslots, the equivalent difference in Mbps reaches 660 Mbps. Moreover, when K receives the value 6, the difference becomes greatest and reaches 1620 Mbps, while the equivalent difference in mean time delay does not surpass 0.7 timeslots (34.02 for POSA and 34.71 for CS-POSA). So, it can be concluded that CS-POSA does not lack significantly in time delay, that means that the improvement that it brings to the network in the context of throughput is stable and without extra time network burden.

Throughput vs. load results

The results from the comparison of the two algorithms are shown in the figure that follows (Fig. 23). The figure presents the behavior of the algorithms, i.e., the relation between throughput and load, changing the values of the workload of the network, i.e., of K. The values of K that are tested are: 2, 4, 6, 8, 10, 20, 30, 40, 50, 100, and 200. The number of the nodes is 24, while the available channels are 12. The speed of the line has been defined at 2.4 Gbps.

It must be mentioned that while the workload of the network is increased, the throughput is decreased. For example, when K equals 2, the throughput equals 22.21 Gbps. When K equals 10, the throughput equals approximately 21.7 Gbps. Finally, when K equals 200, the throughput is decreased reaching 19.13 Gbps. This phenomenon is not often met in the category of the networks examined. Nevertheless, it appears in both algorithms examined, OIS [11], POSA [12],

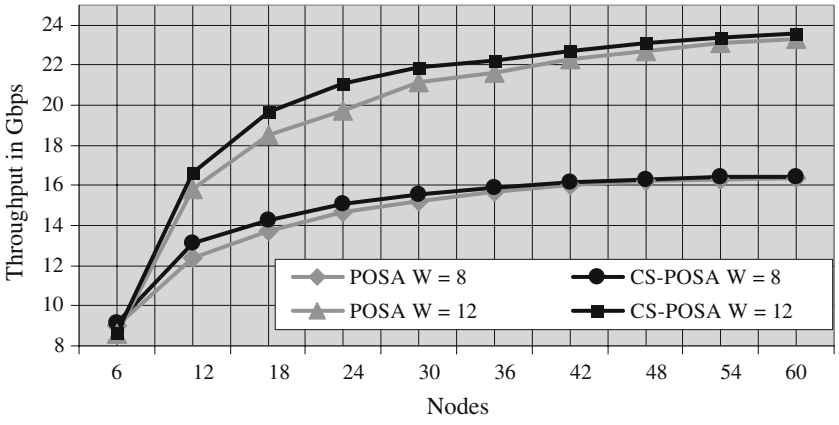

Fig. 20 Throughput (2.4 Gbps)

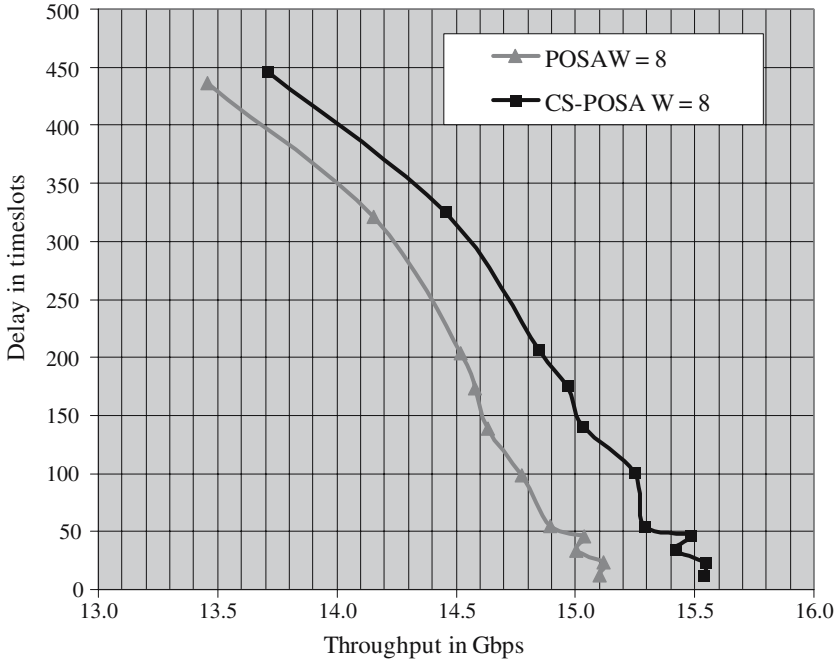

Fig. 21 Throughput vs. delay (2.4 Gbps)

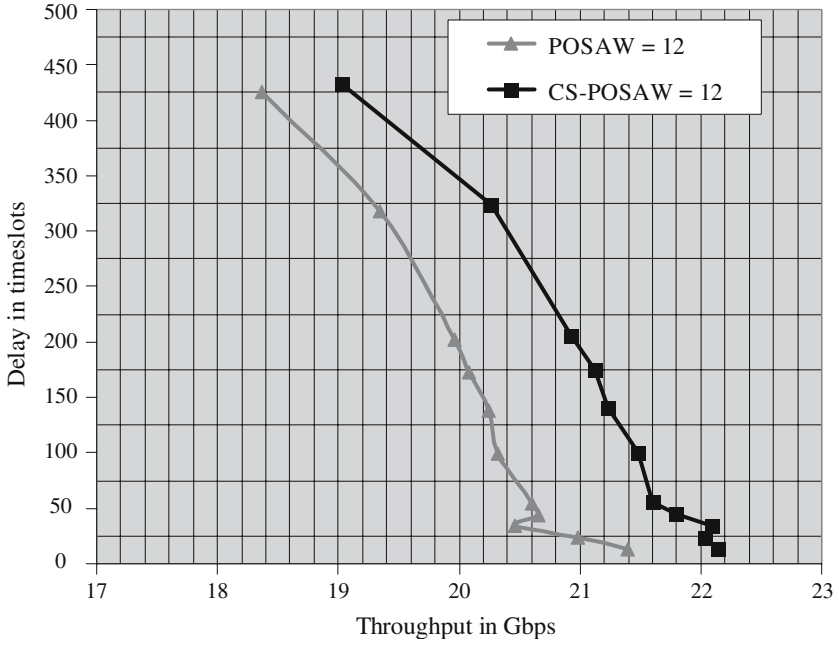

Fig. 22 Throughput vs. delay (2.4 Gbps)

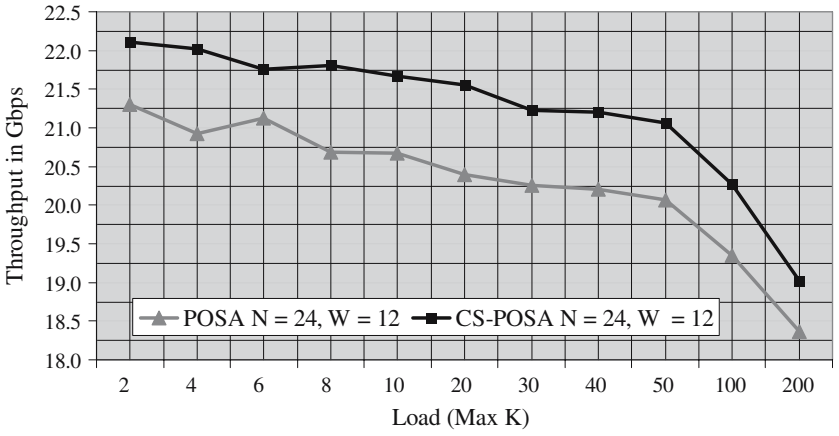

Fig. 23 Throughput vs. load (2.4 Gbps)

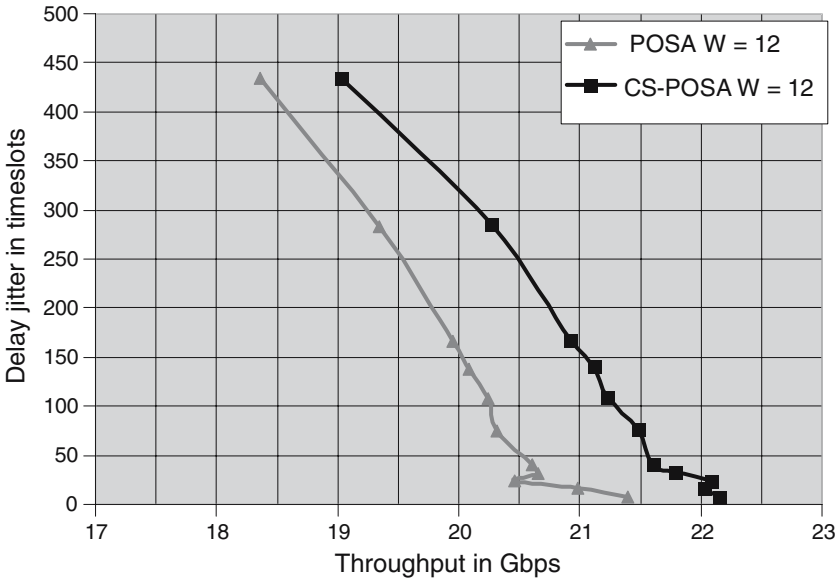

Fig. 24 Throughput vs. delay jitter (2.4 Gbps)

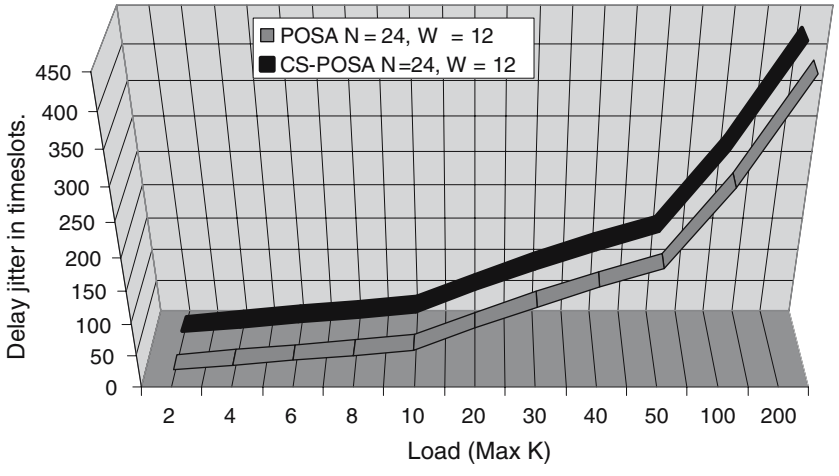

Fig. 25 Delay jitter vs. load (2.4 Gbps)

owing to the architecture of the protocols. When the workload is increased means that the sizes of packets that arrive at the nodes in order to be transmitted, are actually increased. This is denoted with the increase of the maximum value of K. When K is increased, it is difficult for the scheduling algorithm to find open space in the constructed scheduling matrix. If there was an open space of nine slots in the constructed scheduling matrix and the packet arrived was of 10 timeslots size-duration of transmission, then the algorithm could not break it in pieces. It could then place it at the end of the matrix where there would be available space for a packet of 10 timeslots. This leads to a decrease of the channel utilization as the unused timeslots are increased and the throughput is decreased.

Throughput vs. delay jitter results

The results from the comparison between the two algorithms are presented in the Fig. 24. The behavior of the two algorithms is presented, i.e., the relation throughput-delay jitter, altering the values of the workload of the network i.e., of K. The values of K that are tested are: 2, 4, 6, 8, 10, 20, 30, 40, 50, 100, and 200. The number of the nodes is 24, while the available channels are 12. The line speed has been set in 2.4 Gbps.

In the graph, it is obvious that there is a constant difference between the algorithms in the context of throughput as the time delay is increased. In other words, it can be observed that for each value of K, both algorithms have almost the same delay jitter, while CS-POSA is seen improved in the context of throughput. The maximum difference in delay jitter between the two algorithms is observed when K ranges from zero to 100 and is approximately equal to 2.57 timeslots, in favor of POSA, whereas the difference in throughput is 917 Mbps, in favor of CS-POSA. The minimum difference is observed when K ranges from 0 to two and is approximately equal to 0.08 timeslots, in favor of POSA whereas the difference in throughput is 752 Mbps, in favor of CS-POSA. So, it can be concluded that CS-POSA does not lack significantly in delay jitter, which means that the improvement that it brings to the network in the context of throughput is stable.

Delay jitter vs. load results

The results from the comparison of the two algorithms are shown in the Fig. 25. The figure presents the behavior of the algorithms, i.e., the relation between delay jitter and load, changing the values of the workload of the network i.e., of K. The values of K that are tested are: 2, 4, 6, 8, 10, 20, 30, 40, 50, 100, and 200. The number of the nodes is 24, while the available channels are 12. The speed of the line has been defined at 2.4 Gbps. It can be observed that for each value of K, both algorithms have almost the same delay jitter. The two algorithms do not differ greatly, since for each value of the workload, CS-POSA precedes POSA without a significant delay-jitter. The maximum difference in delay jitter between the two algorithms is observed when K ranges from 0 to 100 and is approximately equal to 2.57 timeslots, in favor of POSA. The minimum difference is observed when K ranges from zero to 2 and is approximately equal to 0.08 timeslots, in favor of POSA.

Conclusions

This paper has presented an improved protocol that belongs in the category Broadcast-and-Select, with a star coupled WDM architecture. The protocol is collision-free, pretransmission coordination-based. It uses a learning algorithm, CS-POSA, to predict the requests of the nodes for the following data frame. CS-POSA has a advantage of being programmed to comprehend the workload of each individual node separately so that it can construct a more effective scheduling matrix based on which it transmits the data to the destination nodes. It gets from the predictor the constructed scheduling matrix for the following data frame and simultaneously shifts the order of control and service of the nodes, starting from the one with the additively most requests and finishing with the one with the least. All the above occur at the same time with the transmission of the packets of the current transmission frame. Conclusively, CS-POSA maintains all the positive and remarkable aspects of the OIS and POSA algorithms and at the same time it uses a different policy for the evaluation and service of the requests of the nodes. In this way it improves not only the schedule utilization and the throughput of the network, but also the mean time delay in relation to the throughput. So, it is a reliable solution in the context of network throughput, without extra time burden or extra hardware implementation.

References

1. Papadimitriou, G.I., Papazoglou, Ch, Pompotrsis, A.S.: Optical switching: switch fabrics, techniques, and architectures, IEEE/OSA J. Lightwave Technol. **21**(2), 384–405 (2003)
2. Brackett, C.A.: Dense wavelength division multiplexing network: principles and applications, IEEE J. Select. Areas Commun. **8**, 948–964 (1990)
3. Stern, T.E., Bala, K.: Multiwavelength Optical Networks, Addison-Wesley, Reading, MA, (1999)
4. Green, P.: Progress in optical networking. IEEE Commun. Magaz. **39**(1), 54–61 (2001)
5. Tsukada, M., Keating, A.J.: Broadcast and select switching system based on optical time-division multiplexing (OTDM) technology. IEICE Trans. Commun. **E82-B**(2), 335–343 (1999)
6. Papadimitriou, G.I., Miliou, A.N., Pomportsis, A.S.: OCON: an optically controlled optical network. Computer Commun. **22**, 811–824 (1998)
7. Papadimitriou, G.I., Miliou, A.N., Pomportsis, A.S.: Optical logic circuits : a new approach to the control of fibre optic LANs. In: Proceedings IEEE 23rd Annual Conference on Local Computer Networks (LCN'98), Boston, Massachusetts, pp. 326–335, (1998)
8. Sivalingam, K.M., Dowd, P.W.: A multi-level WDM access protocol for an optically interconnected multiprocessor system. IEEE/OSA J. Lightwave Technol. **13**(11), 2152–2167 (1995)
9. Sivalingam, K.M., Wang, J.: Media access protocols for WDM networks with on-line scheduling. IEEE/OSA J. Lightwave Technol. **14**(6), 1278–1286 (1996)
10. Sivalingam, K.M., Wang, J., Wu, X., Mishra, M.: Improved on-line scheduling algorithms for optical WDM networks, DIMACS Workshop on Multichannel Optical Networks, New Brunswick, NJ, pp. 43–61 (1998)
11. Sivalingam, K.M., Wang, J., Wu, J., Mishra, M.: An interval-based scheduling algorithm for optical WDM star networks. Photonic Netw. Commun. **4**(1), 73–87 (2002)

12. Johnson, E., Mishra, M., Sivalingam, K.M.: Scheduling in optical WDM networks using hidden Markov chain based traffic prediction. Photonic Netw. Commun. **3**(3), 271–286 (2001)
13. Papadimitriou, G.I., Tsimoulas, P.A., Obaidat, M.S., Pomportsis, A.S.: Multiwavelength Optical LANs, Wiley, New York, (2003)
14. Habbab, I.M.I., Kavehrad, M., Sundberg, C.W.: Protocols for very high-speed optical fibre local area networks using a passive star topology. IEEE/OSA J. Lightwave Technol. **LT-5**(12), 1782–1794 (1987)
15. Shi, H., Kavehrad, M.: Aloha/slotted-CSMA protocol for a very high-speed optical fibre local area network using passive star topology. Proceedings IEEE INFOCOM'91, Bal Harbour, Florida, USA, vol. 3. pp. 1510–1515, (1991)
16. Chen, M.S., Dono, N.R., Ramaswami, R.: A media-access protocol for packet-switched wavelength-division metropolitan area networks. IEEE J. Select. Areas Commun. **8**(6), 1048–1057 (1990)
17. Chlamtac, I., Fumagalli, A.: QUADRO-Star: high performance optical WDM star networks. Proceedings IEEE Globecom'91, Phoenix, Arizona, USA, vol. 42. pp. 2582–2590 (1991)
18. Humblet, P.A., Ramaswami, R., Sivarajan, K.N.: An efficient communication protocol for high-speed packet switched multichannel networks. IEEE J. Select. Areas Commun. **11**(4), 568–578 (1993)
19. Ito, Y., Urano, Y., Muratani, T., Yamaguchi, M.: Analysis of a switch matrix for an SS/TDMA system. Proceedings of the IEEE, **65**(3), 411–419 (1977)
20. Borella, M.S., Mukherjee, B.: Efficient scheduling of nonuniform packet traffic in a WDM/TDM local lightwave network with arbitrary transceiver tuning latencies. IEEE J. Select. Areas Commun. **14**(5), 923–934 (1996)
21. Azizoglou, M., Barry, R.A., Mikhtar, A.: Impact of tuning delay on the performance of bandwidth-limited optical broadcast networks with uniform traffic. IEEE J. Select. Areas Commun. **14**(5), 935–944 (1996)
22. Ferrari, D.: Distributed delay jitter control in packet-switching internetworks. J. Internetwork Res. Exp. **4**(1), 1–20 (1993)

Panagiotis G. Sarigiannidis received the B.S. degree in computer science from the Department of Informatics of Aristotle University, Thessaloniki, Greece, in 2001. He is currently working toward the Ph.D. degree in optical networks at the same university. His research interests include optical networks and optical switching.

Georgios I. Papadimitriou received the Diploma and Ph.D. degrees in Computer Engineering from the University of Patras, Greece in 1989 and 1994, respectively. From 1989 to 1994 he was a Teaching Assistant at the Department of Computer Engineering of the University of Patras and a Research Scientist at the Computer Technology Institute, Patras, Greece. From 1994 to 1996 he was a Postdoctorate Research Associate at the Computer Technology Institute. From 1997 to 2001, he was a Lecturer at the Department of Informatics, Aristotle University of Thessaloniki, Greece. Since 2001 he is an Assistant Professor at the Department of Informatics, Aristotle University of Thessaloniki, Greece. His research interests include optical networks, wireless networks, high speed LANs and learning automata. Prof. Papadimitriou is Associate Editor of six scholarly journals, including the IEEE Transactions on Systems, Man and Cybernetics-Part C, the IEEE Transactions on Broadcasting, the IEEE Communications Magazine and the IEEE Sensors Journal. He is co-author of the books "Multiwavelength Optical LANs" (Wiley, 2003) and "Wireless Networks" (Wiley, 2003) and co-editor of the book "Applied System Simulation" (Kluwer, 2003). He is the author of more than 120 refereed journal and conference papers. He is a Senior Member of IEEE.

Andreas S. Pomportsis received the B.S. degree in physics and the M.S. degree in electronics and communications, both from the University of Thessaloniki, Thessaloniki, Greece, and the Diploma in electrical engineering from the Technical University of Thessaloniki, Thessaloniki, Greece. In 1987, he received the Ph.D. degree in computer science from the University of Thessaloniki. Currently, he is a Professor at the Department of Informatics, Aristotle University, Thessaloniki, Greece. He is co-author of the books Wireless Networks (New York: Wiley, 2003) and Multiwavelength Optical LANs (New York: Wiley, 2003). His research interests include computer networks, learning automata, computer architecture, parallel and distributed computer systems, and multimedia systems.