INTERNATIONAL JOURNAL OF COMMUNICATION SYSTEMS
Int. J. Commun. Syst. (2008)
Published online in Wiley InterScience (www.interscience.wiley.com). DOI: 10.1002/dac.929

Clustering-based scheduling: A new class of scheduling algorithms for single-hop lightwave networks

Sophia G. Petridou, Panagiotis G. Sarigiannidis, Georgios I. Papadimitriou*,† and Andreas S. Pomportsis

Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece

SUMMARY

In wavelength division multiplexing (WDM) star networks, the construction of the transmission schedule is a key issue, which essentially affects the network performance. Up to now, classic scheduling techniques consider the nodes' requests in a sequential service order. However, these approaches are static and do not take into account the individual traffic pattern of each node. Owing to this major drawback, they suffer from low performance, especially when operating under asymmetric traffic. In this paper, a new class of scheduling algorithms for WDM star networks, which is based on the use of clustering techniques, is introduced. According to the proposed Clustering-Based Scheduling Algorithm (CBSA), the network's nodes are organized into clusters, based on the number of their requests per channel. Then, their transmission priority is defined beginning from the nodes belonging to clusters with higher demands and ending to the nodes of clusters with fewer requests. The main objective of the proposed scheme is to minimize the length of the schedule by rearranging the nodes' service order. Furthermore, the proposed CBSA scheme adopts a prediction mechanism to minimize the computational complexity of the scheduling algorithm. Extensive simulation results are presented, which clearly indicate that the proposed approach leads to a significantly higher throughput-delay performance when compared with conventional scheduling algorithms. We believe that the proposed clustering-based approach can be the base of a new generation of high-performance scheduling algorithms for WDM star networks. Copyright © 2008 John Wiley & Sons, Ltd.

Received 27 November 2007; Revised 12 January 2008; Accepted 12 January 2008

KEY WORDS: WDM star networks; scheduling; reservation; prediction; clustering

1. INTRODUCTION

Nowadays, the spread of Internet increases dramatically the number of end users who, at the same time, become more demanding in terms of the provided capacity. Three main technologies

*Correspondence to: Georgios I. Papadimitriou, Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece.
†E-mail: gp@csd.auth.gr

compete to cover the ever-growing users' needs, namely the copper, the wireless and the optical technology [1]. Given that copper and wireless technologies provide users with limited capacity, it seems that optical networking is the most promising technology for coping with the aforementioned demands and meeting both present and future needs due to the potentially unlimited capacity of optical fibers [2]. However, if we wish to utilize the optical fiber in a cost-effective manner, it is useful to share all of its huge capacity among several network nodes. Wavelength division multiplexing (WDM) technique offers an excellent way of exploiting the huge bandwidth of optical fibers by introducing concurrency among multiple users transmitting at electronically feasible rates [3].

WDM networks can be classified into four categories: point-to-point link networks, broadcast-and-select networks, wavelength-routed networks and passive optical networks [4]. In turn, each of these can be further classified as either single-hop or multi-hop network. The star topology using single-hop, broadcast-and-select architecture seems to dominate in local area networks (LANs). In this topology, network nodes are connecting to a passive star coupler via two-way fibers. The star coupler is located at the center of the network and its role is to combine the incoming optical signals from each node and then to send it to all nodes. Subsequently, each node has to use its receiver to select the desired wavelength for data reception. In general, network nodes can transmit and receive data on any of the available channels employing one or more fixed or tunable transmitter(s) (FT or TT) and one or more fixed or tunable receiver(s) (FR or TR) [5].

An important issue in such WDM networks is to be specified the way that nodes transmit on the available channels [6]. Thus, a media access control (MAC) protocol is needed to coordinate the nodes' data transmission and prevent collisions [7]. As depicted in Figure 1, MAC protocols can be characterized either as pre-allocation-based or as pre-transmission-coordination-based according to the existence of a channel which is called control channel [1]. More specifically, in the pre-allocation-based protocols there is no control channel and thus wavelengths are pre-allocated to the transmitters or receivers, whereas in the pre-transmission-coordination-based

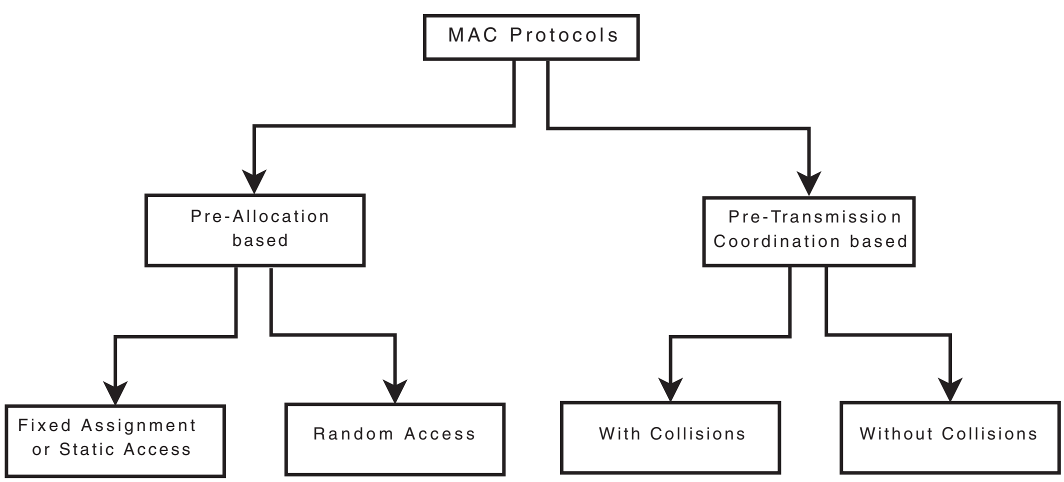


Figure 1. Classification of MAC protocols.

DOI: 10.1002/dac

protocols a control channel is used for nodes' coordination before their actual data transmission. Pre-allocation-based protocols are further divided into fixed-assignment or static access and random access protocols, whereas the pre-transmission-coordination-based protocols can be characterized either as with collisions or as without collisions according to whether or not prevent collisions. Representatives of MAC protocols that allow collisions can be found in [8–11], whereas References [12–15] present pre-transmission-coordination-based protocols without collisions.

This work focuses on a special category of pre-transmission-coordination-based protocols in which the transmission coordination is achieved without any control channel (for economic reasons). A typical pre-transmission-coordination-based scheduling algorithm for optical WDM networks is the online interval-based scheduling (OIS) [12] while an extension of OIS, which is based on traffic prediction, is the Predictive Online Scheduling Algorithm (POSA) [14]. Both protocols schedule traffic considering network nodes in a sequential service order. However, sequential scheduling leads to a significant performance degradation in terms of network throughput and mean packet delay, since nodes with short-length requests (few packets) may transmit prior to those with long-length requests. Moreover, the more asymmetric the network traffic, the greater the performance degradation since sequential scheduling fails to separate nodes according to their load.

This work is inspired by the fact that, in practice, the network load is asymmetric and thus there is a need of a scheduling scheme that would take into account the specific demands of each node. Thus, in this paper we present a new pre-transmission coordination scheduling algorithm for optical WDM star networks which is based on the clustering [16, 17] of network nodes. The proposed Clustering-Based Scheduling Algorithm (CBSA) organizes the network's nodes into groups according to the number of their requests per channel. In general, clustering aims at creating groups of items, i.e. clusters on the basis that items assigned to the same cluster are 'similar' to each other and 'dissimilar' to the nodes belonging to other clusters [16]. In our framework, CBSA manages to separate nodes according to their load and thus defines their transmission priority beginning from the nodes belonging to the cluster with greater demands and ending to the nodes of cluster with fewer requests. In this way, we decrease both the unused time slots as well as the schedule length and as a result the network performance is significantly upgraded without aggravating the time complexity of the scheduling algorithm.

Data clustering is a common technique for data analysis, which is used in many fields, including Web data mining [17, 18], image analysis [19] and bioinformatics [20]. Especially on the Web, many research efforts have focused on grouping users that present similar access behavior [21, 22]. Collecting information about users' behavior and extracting their usage patterns can be quite important for providing dynamic Web content, for effective Web site structuring and management as well as for improving specific applications such as e-commerce via pages' caching and prediction.

The remainder of this paper is organized as follows. Section 2 provides the network background while Section 3 presents related scheduling algorithms. Clustering background is given in Section 4 while our new scheduling algorithm is described in Section 5. Section 6 provides details about the traffic model we use, while Section 7 presents a graphical analysis of the clustering results. In Section 8 extensive simulation results are presented which indicate the superiority of the proposed CBSA scheme over the POSA one. Conclusions and future work insights are given in Section 9.

DOI: 10.1002/dac

2. NETWORK BACKGROUND

This section provides background issues that are related to the network architecture as well as assumptions about the transmission and scheduling process. Network notation summary is given in Table I.

2.1. Network architecture

Consider a local area WDM single-hop network with broadcast-and-select architecture. This network consists of n nodes, which are connected in a passive star coupler via a two-way optical fiber, and w data channels (wavelengths), where $n \geqslant w$, which are of the same capacity. According to Table I, $U = \{u_1, \ldots, u_n\}$ denotes the set of network's nodes while $\Lambda = \{\lambda_1, \ldots, \lambda_w\}$ indicates the set of data channels. Although there is no separate control channel for coordination, the proposed protocol is still pre-transmission-coordination-based, since the set of w data channels are used for both control and data packets. Thus, each node may transmit data on different channels using a TT, while it receives packets in a dedicated channel, also known as home channel, as depicted in Figure 2.

However, since the number of nodes is greater than the number of channels, it holds that n/w nodes share the same home channel. In general, in an n-node optical broadcast-and-select network, the most effective communication structure is achieved when each node has a TT and a TR (TT–TR system) in combination with $w = n$ channels, i.e. one channel per node. In this way, each node would have its own unique channel and the protocol would become collision free. However, TT–TR systems are very difficult to be applied in practice due to network size scalability reasons as well as due to the transceivers high cost [6]. Furthermore, technological constraints dictate that the number of WDM channels supported in a fiber should be limited, e.g. as many as 160 in some systems [23]. Thus, the solution of a TT and a FR system (TT–FR) seems to be suitable under the current technological and financial constraints.

2.2. Protocol issues

In the above TT–FR implementation, it is clear that two or more nodes may transmit packets on the same data channel simultaneously causing channel collision. In this case, the packets are destroyed while the nodes are informed and retransmit the corrupted data leading to bandwidth waste and

Table I. Network notation.

Symbol	Description
n, w	Number of nodes and channels
$U = \{u_1, \ldots, u_n\}$	The set of network's nodes
$\Lambda = \{\lambda_1, \ldots, \lambda_w\}$	The set of network's channels
D	The $n \times w$ demand matrix
k	The upper bound of nodes' requests
t	Schedule's length in time slots
$L = \{l_1, \ldots, l_t\}$	The set of time slots
S	The $w \times t$ scheduling matrix

DOI: 10.1002/dac

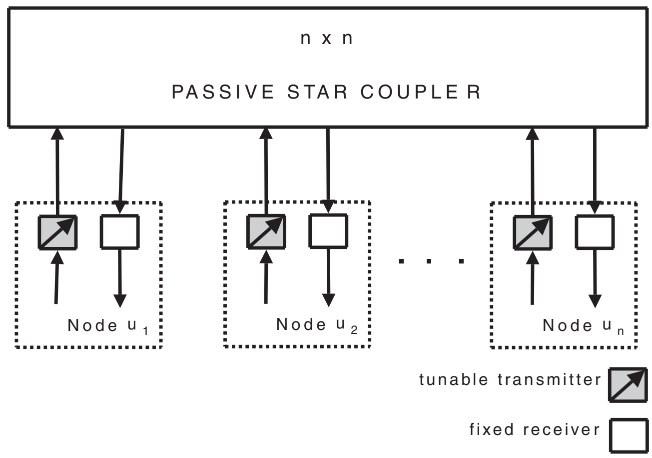


Figure 2. Network topology.

extra time delays. Thus, a MAC protocol is needed to support a set of access rules aiming at preventing collisions and specifying the way that nodes transmit on the available channels.

In general, the design of a collision-free protocol requires a synchronization mechanism. In our case, this synchronization mechanism is applied by a distributed scheduling algorithm that operates on all nodes simultaneously. The proposed scheduling algorithm is based on global status information which, in our case, is an $n \times w$ demand matrix D, where $d(i, j)$ element, $i = 1, \ldots, n$ and $j = 1, \ldots, w$, indicates the number of data packets that the node u_i destines to the channel λ_j.

The proposed protocol is based on the following assumptions [12–15]:

1. Transmission is organized into frames, where each frame consists of a reservation (or control) phase and a data phase.
2. Nodes receive the control packets on their corresponding home channel using their FRs.
3. The reservation phase consists of n time slots during which the n nodes broadcast their requests to all channels using their TTs.
4. Nodes' requests are formed as variable-length messages consisting of one or more fixed-length data packets.
5. Time is divided into time slots and each fixed-length data packet is transmitted in time equal to a time slot.
6. Each node consists of w queues and stores its messages on the channel queue, which corresponds to the destination node's home channel.
7. The proposed scheme produces the $w \times t$ scheduling matrix S, where t denotes the length of the schedule in time slots.
8. Each $s(i, j)$ element, $i = 1, \ldots, w$ and $j = 1, \ldots, t$, represents the node that transmits on channel λ_i during the time slot l_j.

Based on the above, it holds that at the beginning of each frame all nodes run the same distributed scheduling algorithm based on the same global information, i.e. the demand matrix D. Thus, the proposed scheme manages to define a collision-free order of transmissions and receptions for the next frame.

DOI: 10.1002/dac

3. RELATED SCHEDULING ALGORITHMS

As discussed in Section 1, the main differentiation between MAC protocols is based on the existence of a control channel. This separate wavelength is used for nodes' coordination before actual transmission, i.e. during the control phase. If a protocol uses one control channel (at least), then it is called a pre-transmission coordination-based protocol, whereas when no control channel is used a MAC protocol should pre-allocate wavelengths to the transmitters or receivers. This work focuses on a special category of pre-transmission-coordination-based protocols in which the transmission coordination is achieved without any control channel (for economic reasons). Hence, we assume that coordination is accomplished by transmitting control packets over the data channels in a TDM fashion.

The proposed scheme is based on two pre-transmission-coordination-based MAC protocols, namely OIS [12] and POSA [14]. OIS and POSA are characterized as online schemes and prevent channel collisions by employing a distributed scheduling approach. OIS protocol incorporates online scheduling on the basis that the scheduling algorithm begins the schedule construction once the first node's control packet is received. In other words, OIS calculates the schedule when the first row of the demand matrix D is known. It has low time complexity and it is very simple in practice.

More specifically, the OIS algorithm maintains two sets of intervals, one for each channel and a second for the node whose reservation is currently being scheduled. Let us suppose that the list of the available intervals for a certain channel λ_j at a time point contains the time slots $[3,\ldots,8]$ and $[14,\ldots,\infty)$. This implies that time slots $1,2$ and $[9,\ldots,13]$ have already been assigned to one or more nodes and, thus, they are blocked for any other node for the current frame. Furthermore, for each node whose request is being processed, the algorithm maintains an additional list of intervals. This second list represents the time slots that have not yet been assigned to this node. If we assume that the list of intervals for a certain node n_i contains the time slots $[8,\ldots,11]$, $[15,\ldots,20]$ and $[25,\ldots,\infty)$, then it holds that the node n_i has already been scheduled to transmit at time slots $[1,\ldots,7]$, $[12,\ldots,14]$ and $[21,\ldots,24]$. Given the available intervals of channel λ_j and node n_i, a possible request $d(i,j)=3$ would be scheduled during the time slots $[15,\ldots,17]$. Algorithm 1 provides the OIS' pseudo-code.

The OIS protocol produces a schedule for transmitting the requests of demand matrix D and runs in time linear on the number of nodes n, i.e. $O(nw^2k)$, where k is the upper bound of nodes' requests on each channel. The fact that OIS is a simple algorithm including simple and fast procedures without decomposing the demand matrix D, as other protocols of this category (HRP/TSA [13]), offers powerful advantages as far as its execution is concerned. The main drawback of OIS is that it requires a lot of construction time for the final schedule of each frame, since the data transmission should wait for the completion of the schedule of each frame.

POSA tries to eliminate the possible delay introduced by the scheduling computation between the control and data phases of each frame to reduce the waiting time of the nodes due to the control phase. POSA is actually an extension of OIS which is based on traffic prediction according to the history of recent reservation requests. POSA assumes that the reservation information of each control phase constitutes the input to a mechanism consisting of $n\times w$ predictors whose operation is mainly based on the hidden Markov chain model. More specifically, after an initial period, i.e. learning phase, the protocol learns the traffic pattern by maintaining a reservation history and then switches into a prediction phase. During the prediction phase, POSA calculates the schedule based on the predicted reservations (of the previous control phase), instead of the actual reservations of

DOI: 10.1002/dac

the current control phase. As a result, POSA offers more computational time for the construction of the next schedule; thus, the schedule is available at the beginning of the data phase.

Algorithm 1 The OIS pseudo-code

Input: A set U of n nodes whose packets' requests on each of the w channels organized in an $n \times w$ demand matrix D and the upper bound on nodes' requests k.
Output: The scheduling matrix S.

Begin frame
Initialize w intervals, i.e. one for each channel
for $i := 1$ to n **do**
  Initialize the set of intervals of node u_i
  for $j := 1$ to w **do**
    Find a suitable time space for the request of node u_i on channel λ_j starting from the beginning of frame, i.e. time slot 0 so that there is no other node scheduled during this space on this channel and there is no schedule for node u_i on any other channel
    Update the set of interval of channel λ_j
  end for
  Update the set of intervals of node u_i
end for
End frame

It is important to mention that during the learning phase which lasts, for example, v frames, the POSA protocol operates similar to OIS. Furthermore, the prediction mechanism of POSA should be accurate enough otherwise it leads the scheduling algorithm to false output. According to Johnson *et al.* [14], it has been found that in 70% of the predictions, the percentage of failure was less than 20%, regardless of the number of nodes n, channels w or the upper bound of nodes' requests k. Therefore, POSA does not affect the schedule process. Its goal is to minimize the calculation time of the scheduling process. Thus, POSA protocol does not add extra complexity to the system, since its prediction mechanism runs linearly with the increase of nodes or channels: $O(k+1+v)(nw)$.

4. CLUSTERING BACKGROUND

A clustering process aims at creating groups of similar items, e.g. nodes that exhibit similar traffic patterns. Thus, in our framework, clustering considers the set of nodes $U = \{u_1, \ldots, u_n\}$ on the basis of the $n \times w$ demand matrix D whose $d(i, j)$ element, $i = 1, \ldots, n$ and $j = 1, \ldots, w$, indicates the number of data packets that the node u_i destines to channel λ_j. Given that each node u_i is represented by the row i of the demand matrix D, this row can be denoted as a multivariate vector consisting of w values as follows:

$$D(i, :) = (d(i, 1), \ldots, d(i, w))$$

The vector $D(i, :)$ represents the demand vector or traffic pattern of node u_i.

Thus, according to Table II, we can define that a clustering Cl of U is a partition of U into noc disjoints sets, i.e. clusters $C_1, \ldots, C_{\text{noc}}$, that is, $\bigcup_{i=1}^{\text{noc}} C_i = U$ and $C_i \cap C_j = \emptyset$ for all $i \neq j$.

DOI: 10.1002/dac

Table II. Clustering notation.

Symbol	Description
Cl	Clustering process
noc	Number of clusters
C_j	Cluster, $j=1,\dots,\mathrm{noc}$
$f(u_i, C_j)$	Function membership of node u_i to cluster C_j, $i=1,\dots,n$
c_j	Cluster representative
Means	The $\mathrm{noc}\times w$ cluster representatives' demand matrix
d_{E}	Nodes distance over channels
J	Criterion function

The noc clusters $C_1,\dots,C_{\mathrm{noc}}$ consist of $|C_1|,\dots,|C_{\mathrm{noc}}|$ members (i.e. nodes), respectively. Nodes assigned to the same cluster are 'similar' to each other and 'dissimilar' to the nodes belonging to other clusters in terms of their packets' requests per channel. The membership of a node u_i, where $i=1,\dots,n$, to a cluster C_j, where $j=1,\dots,\mathrm{noc}$ is defined by the function f as follows:

$$f(u_i, C_j)=\begin{cases}1 & \text{if } u_i\in C_j\\ 0 & \text{otherwise}\end{cases}$$

Based on the above, it is apparent that similarity is fundamental to the definition of a cluster. Thus, a measure of the similarity between two patterns (in our case traffic patterns) is essential to most clustering approaches. In practice, it is most common to calculate the dissimilarity between two patterns using a distance measure. However, because of the variety of distance measures, patterns' representation plays a major role to the selection of distance measure [16]. Conventionally, patterns are represented as vectors whose values can be either quantitative (continuous values, e.g. weight, discrete values, e.g. the number of visits of a Web user or interval values, e.g. the duration of an event) or qualitative (nominal, e.g. 'red' or ordinal, e.g 'cool').

Since our nodes' patterns are represented as vectors with discrete values, we will focus on the squared Euclidean distance[‡] which is a well-known and widely used distance measure in the vector-space model [16, 18]. Therefore, the evaluation of the dissimilarity between two nodes can be expressed by the distance of their demand vectors. Thus, $d_{\mathrm{E}}(u_x, u_y)$, where $u_x, u_y\in U$, denotes the squared Euclidean distance of the nodes' demand vectors $D(x,:)$ and $D(y,:)$:

$$d_{\mathrm{E}}(u_x, u_y)=\|D(x,:)-D(y,:)\|^2$$

Let us consider an arbitrary cluster C_j, where $j=1,\dots,\mathrm{noc}$, of the set U. The representation of the cluster C_j when clustering process Cl is applied to it collapses the nodes belonging to C_j into a single point (e.g. the mean value that does not correspond to an existing node). This point is called cluster's representative c_j (also known as centroid) since each node $u_i\in C_j$ is represented

‡The squared Euclidean distance uses the same equation as the Euclidean distance, but does not take the square root. For two points $P=(p_1,\dots,p_n)$ and $Q=(q_1,\dots,q_n)$ in n-space their squared Euclidean distance is defined as $\|p_i-q_i\|^2$.

DOI: 10.1002/dac

by c_j. Given the demand vectors of $u_i \in C_j$, the demand vector of c_j is defined as follows:

$$Means(j,:) = \frac{1}{|C_j|} \sum_{i=1}^{n} f(u_i, C_j) * D(i,:), \quad j = 1, \ldots, \mathrm{noc}$$

Since both $D(i,:)$ and $Means(j,:)$ are vectors, their dissimilarity is measured by their squared Euclidean distance $d_{\mathrm{E}}(u_i, c_j)$. Considering all clusters, the clustering process is guided by the *criterion function* J, which is defined to be the sum of distances over all channels between each node and the representative of the cluster that the node is assigned to

$$J = \sum_{j=1}^{\mathrm{noc}} \sum_{u_i \in C_j} d_{\mathrm{E}}(u_i, c_j)$$

Based on the above, we can define the network nodes clustering as follows: Given a network with n nodes whose packets' requests on each of the w channels are organized in an $n \times w$ demand matrix D, the integers noc and k, and the criterion function J, find a Cl clustering of U into noc clusters such that the J is minimized.

For the purpose of our clustering, we employed the well-known and widely used K-means partitional clustering algorithm [24]. K-means classifies a given data set to a certain number of clusters, e.g. noc fixed *a priori*. Although K-means does not provide approximation guarantees, it is widely used because it is efficient and it works very well in practice. K-means algorithm in summary is given n points to be clustered, a distance measure d_{E} to capture their dissimilarity and the number of clusters noc to be created, the algorithm initially selects noc random points as clusters' centers and assigns the rest of the $n - \mathrm{noc}$ points to the closest cluster center (according to d_{E}). Then, within each of these noc clusters the cluster representative (also known as centroid or mean) is computed and the process continues iteratively with these representatives as the new clusters' centers, until convergence.

5. THE PROPOSED CLUSTERING BASED SCHEDULING ALGORITHM

Section 3 makes clear that OIS and POSA protocols consider the nodes' requests in a sequential service order which, as it has already discussed, does not take into account the specific demands of each node. Thus, the above protocols suffer from low performance especially when operating under asymmetric traffic. The goal of the proposed novel algorithm is to reduce the schedule length by modifying the service order of the network's nodes. For this purpose, CBSA employs clustering techniques to separate nodes into groups according to their load. Discovering such groups, i.e. clusters, it manages to rearrange the nodes' service order and, thus, to schedule them beginning from the nodes belonging to clusters with higher demands and ending to the nodes of clusters with fewer requests. The goal of this rearrangement is to minimize the time gaps or idle time slots of the schedule and as a result to improve the network's performance. Since, a typical metric of the efficiency of the schedule is the channel utilization (i.e. the percentage of the demanded slots over the total slots of the scheduling matrix S), the fewer the idle time slots the higher the channel utilization and, therefore, the higher the network throughput.

As far as the scheduling technique is concerned, the CBSA employs OIS due to its simplicity. Furthermore, the prediction mechanism of POSA is adopted to minimize the computational time between the reservation and the data phase. Certainly, CBSA could also operate without a prediction

DOI: 10.1002/dac

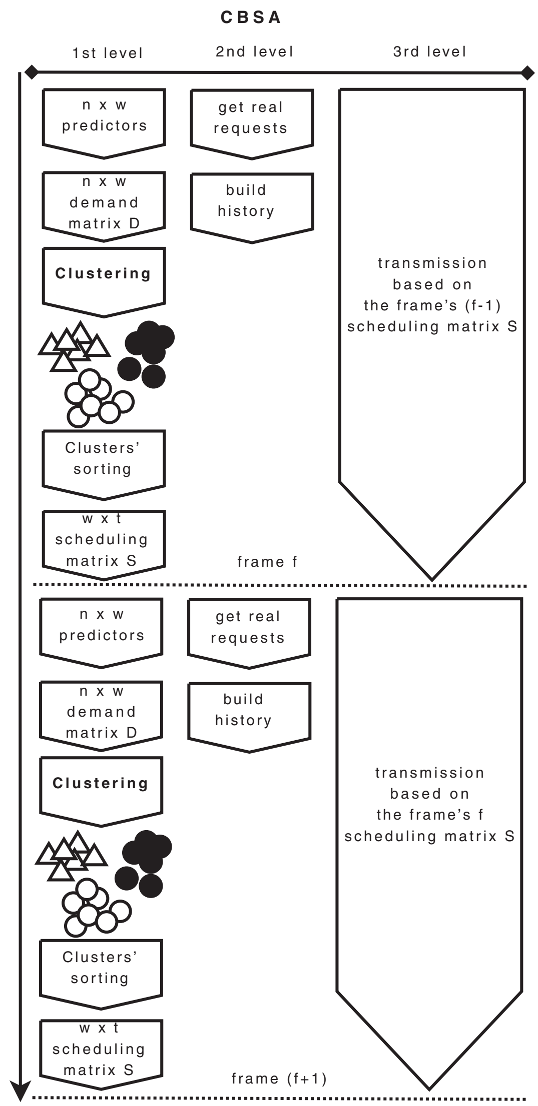


Figure 3. The CBSA overview.

mechanism, if, for example, it is impossible to predict the traffic pattern accurately. It is crucial for the optical domain to keep low the computational complexity of the scheduling algorithm. Hence, the proposed scheme does not surpass the OIS algorithm in complexity, while it keeps the same complexity of the POSA's prediction mechanism. Furthermore, CBSA is scalable, due to the fact that it runs in time linear with the number of nodes. Finally, the CBSA's performance is superior than POSA, since the produced schedule matrix allows a better use of the channels regardless of the values of nodes n, channels w and the upper bound of nodes' requests k.

A general overview of the CBSA is depicted in Figure 3. At the beginning of each frame, CBSA operates in parallel on three levels. Let us suppose the frame f of a transmission. On the first level (at the left part of the Figure 3) of frame f, the clustering procedure takes part. More specifically, the prediction mechanism that is implemented using $n\times w$ predictors produces the $n\times w$ (predicted) demand matrix D. The clustering process is applied on this D matrix and defines the nodes' service order. In practice, noc clusters are created and the clusters with greater requests are prioritized. Afterwards, the final scheduling matrix S is constructed based on the rearrangement of the network nodes. This scheduling matrix will be used in the following frame's transmission. On the second level (in the middle part of the Figure 3), CBSA receives the real nodes' requests (during the reservation phase of the current frame). The real requests of the nodes fill the history and, thus, the algorithm is updated according to the actual traffic patterns. Concurrently, on the third level (at the right part of the Figure 3), the transmission of the packets is performed. It is important to mention that the transmission of the current frame is specified by the previous frame's scheduling matrix, instead of the real nodes' requests of the current frame. To summarize, the first level of frame f feeds the third level of frame $f+1$, whereas the second level of frame f feeds the predictors that operate at the beginning of the first level of frame $f+1$.

Algorithm 2 provides a more detailed overview of the first level, which is composed of three steps. During the first step, namely the prediction step, the CBSA employs the prediction mechanism of POSA in order to construct the $n\times w$ demand matrix D. Then, based on this D matrix, the CBSA proceeds to the clustering step. During this second step, the K-means algorithm is applied to the matrix D and a Cl clustering of the nodes' set U is produced. The K-means minimizes the objective function J defined in Section 4. Next, given the Cl as well as the *Means* table, consisting of the cluster representatives' demand vectors $Means(j,:)$, the *SortedMeans* is computed in order that we prioritize the clusters with greater requests. We sort $Means(j,:)$ vectors according to their length, i.e. $|Means(j,:)|$, since a vector's length is more indicative than the sum of its values for revealing the information that CBSA needs, i.e. the vectors with greater requests. For example, given the vectors $Means(1,:)=(1,1,1)$ and $Means(2,:)=(3,0,0)$ of clusters C_1, C_2, their elements sum is 3, whereas their lengths are $\sqrt{3}$ and 3, respectively, which means that the cluster C_2 will have priority over the cluster C_1 in the service order. The calculated *SortedMeans* is then used in order that the network nodes are rearranged. Once the *ClusteredNodes* is formed, the algorithm proceeds to the third step which is called the scheduling step. The goal of function *Schedule* is to form the scheduling matrix S using the same logic as OIS algorithm.

Theorem 1
The CBSA has time complexity $O(nkw^2)$

Proof
During the prediction step the CBSA employs the POSA (line 2) whose time complexity is $O(k+1+v)(nw)$, where n is the number of nodes, w the number of channels, k the upper bound of nodes' requests and v the duration of learning phase in frames. During the clustering step we employ the K-means algorithm (line 4) whose time complexity is $O(n \text{ noc } r)$, where noc is the number of clusters to be created and r the number of iterations that takes the algorithm to converge. However, both noc and r are relatively small compared with the number of nodes n and thus their contribution to the algorithm's complexity can be ignored [16]. Thus, the Cl clustering is computed in time linear on the number of nodes: $O(n)$. The *Quicksort* function (line 5) sorts clusters' representatives in $O(\text{noc}\log(\text{noc}))$ time, whereas the *Arrange* function (line 6) takes time

DOI: 10.1002/dac

$O(n)$ to arrange the n nodes according to the *SortedMeans*. The total time complexity of the clustering step is thus $O(n+\text{noc}\log(\text{noc})+n)$ which becomes $O(n)$. During the scheduling step, the *Schedule* function (line 8) needs $O(nkw^2)$ time [14] to from the scheduling matrix S. As a result, the total complexity of CBSA is $O(k+1+v)(nw)+O(n)+O(nkw^2)=O(nkw^2)$. □

Algorithm 2 The CBSA flow control

Input: A set U of n nodes whose packets' requests on each of the w channels organized in an $n\times w$ demand matrix D, the upper bound on nodes' requests k and the number of clusters noc.

Output: The scheduling matrix S.

1: */*Prediction Step*/*
2: $D=\mathit{Prediction}(n,w)$
3: */*Clustering Step*/*
4: $(\text{Cl},\mathit{Means})=K-\mathit{means}(D,\mathit{noc})$
5: $\mathit{SortedMeans}=\mathit{Quicksort}(\mathit{Means})$
6: $\mathit{ClusteredNodes}=\mathit{Arrange}(\mathit{SortedMeans})$
7: */*Scheduling Step*/*
8: $S=\mathit{Schedule}(\mathit{ClusteredNodes})$

To facilitate the comprehension of the CBSA, consider a network consisting of $n=6$ nodes, namely $U=\{u_1,u_2,u_3,u_4,u_5,u_6\}$, and $w=3$ channels, namely $\Lambda=\{\lambda_1,\lambda_2,\lambda_3\}$, whereas the upper bound of nodes' requests is $k=4$. Then, a 6×3 demand matrix D could be the following:

$$D=\begin{pmatrix}2 & 0 & 2\\ 1 & 3 & 3\\ 2 & 1 & 1\\ 3 & 3 & 3\\ 1 & 2 & 2\\ 2 & 1 & 0\end{pmatrix}$$

Example 1
In the above demand matrix D the fact that $D(5,3)=2$ means that the node identified as u_5 requests two packets on channel λ_3.

Applying the K-means for $\text{noc}=3$ in the above D matrix results in $\text{Cl}=(3,2,3,1,2,3)$ which means that $u_4\in C_1$, $u_2,u_5\in C_2$ while $u_1,u_3,u_6\in C_3$. Given this Cl the cluster representatives' demand matrix *Means* is formed as follows:

$$\mathit{Means}=\begin{pmatrix}3 & 3 & 3\\ 1 & 2.5 & 2.5\\ 2 & 0.7 & 1\end{pmatrix}$$

Sorting *Means* results in providing the following clusters' priority: C_1,C_2,C_3 since $|\mathit{Means}(1,:)|=5.2$, $|\mathit{Means}(2,:)|=3.7$ and $|\mathit{Means}(3,:)|=2.3$. Therefore, the schedule service order defined by

DOI: 10.1002/dac

Table III. The scheduling matrix S produced by CBSA.

	Time slots											
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9	l_{10}	l_{11}	l_{12}
λ_1	u_4	u_4	u_4	u_2	u_5	u_1	u_1	u_3	u_3	u_6	u_6	
λ_2	u_2	u_2	u_2	u_4	u_4	u_4	u_5	u_5	u_6	u_3		
λ_3	u_5	u_5	u_1	u_1	u_3		u_4	u_4	u_4	u_2	u_2	u_2

Table IV. The scheduling matrix S produced by POSA.

	Time slots													
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9	l_{10}	l_{11}	l_{12}	l_{13}	l_{14}
λ_1	u_1	u_1	u_2	u_3	u_3	u_4	u_4	u_4	u_5	u_6	u_6			
λ_2	u_3	u_5	u_5	u_2	u_2	u_2	u_6		u_4	u_4	u_4			
λ_3		u_3	u_1	u_1	u_5	u_5	u_2	u_2	u_2			u_4	u_4	u_4

the CBSA will be $u_4, u_2, u_5, u_1, u_3, u_6$ instead of $u_1, u_2, u_3, u_4, u_5, u_6$ which is formed by the POSA. Tables III and IV depict the scheduling matrix S produced, respectively. Based on these tables, CBSA provides 89% channels' utilization causing 5.3 time slots as mean packet delay while POSA offers 76% channels' utilization with 5.7 time slots mean packet delay.

The aforementioned example is a random one and does not represent the best case. It should be noted that in the worst case the nodes' requests are very similar between each other. In this case, the clustering step of CBSA is of no value, since it creates similar clusters whose prioritizing does not contribute to the creation of the scheduling matrix. Under this worst scenario, the performance of CBSA is similar to that of POSA. Such an experiment was carried out, where the table D was as follows:

$$D = \begin{pmatrix} 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \\ 2 & 2 & 2 \end{pmatrix}$$

The K-means for noc$=3$ produced the clustering $\mathrm{Cl}=(3, 1, 1, 1, 1, 1)$ (the second cluster was empty), while the sorting of *Means* results in providing the following clusters' priority: C_1, C_2, C_3. The proposed service order was formed as $u_2, u_3, u_4, u_5, u_6, u_1$ instead of $u_1, u_2, u_3, u_4, u_5, u_6$, which was of no value. Although CBSA and POSA produced different scheduling matrices regarding the nodes' service order, these matrices are similar in terms of actual requests and, thus, both schemes achieved the same channels' utilization which was as much as 75%, as well as, the same mean packet delay which was as many as 5.9 time slots.

DOI: 10.1002/dac

The proposed scheme considers that all packets are of equal priority. However, since real-time traffic (high-priority packets) represents 25–30% of the Internet traffic, the proposed CBSA can be easily extended to handle real-time traffic, such as audio and video. More specifically, during each frame the nodes include in their control packets the priority information of their data packets, i.e. $p_r = 1$ for high-priority packets and $p_r = 0$ for low-priority packets. Then, high- and low-priority packets are clustered and scheduled separately, with high-priority packets having the privilege of being scheduled prior to low-priority ones. Thus, the proposed scheme succeeds in obtaining significant improvements for real-time traffic, without sacrificing the performance for nonreal-time traffic, such as text, e-mail or file transfer.

6. TRAFFIC MODELING

For the purpose of our experimentation, we used two distinct traffic models. According to the first model, namely the model A or uniform model, it is assumed that the packet arrival process on each of the w queues follows uniform distribution. In practice, each node u_i, where $i = 1, \ldots, n$, may send 0–k packets on each channel during each frame with equal probability.

According to the second model, i.e. model B or asymmetric model, it is assumed that the packet arrival process follows a Poisson process. In general, the Poisson distribution of the number of packets arriving at a specific queue per frame is defined as

$$p(X;\lambda) = \frac{\mathrm{e}^{-\lambda}\lambda^{X}}{X!} \tag{1}$$

where $p(X;\lambda)$ is the probability of X packets being assigned to a specific queue during a specific frame, whereas λ is the expected number of packets being assigned to this queue during this frame.

Based on the above, we proceed to the nodes load patterns' generation defining three classes of nodes to simulate a more realistic environment. More specifically, each node is assigned to a class with equal probability and characterized as light, medium or heavy according to its traffic load. The values of λ for these three classes are defined as $k/4$, $k/2$ and $3k/4$, respectively [14], where k is the upper bound of nodes requests per channel.

7. A GRAPHICAL EVALUATION OF THE CLUSTERING PROCESS

To evaluate the results of the clustering process, we proceeded to a graphical analysis approach, which is appropriate for representing multidimensional data. In general, graphical analysis is very important since it can reveal the underlying structure in a data set [25]. For multidimensional data, advanced multivariate graphical techniques such as Andrews' curves are used to efficiently depict the multidimensional data properties [26]. Each multidimensional observation, e.g. $D(i,:) = (D(i,1), \ldots, D(i,w))$, where $i = 1, \ldots, n$ is transformed into a curve based on the function:

$$f(t) = D(i,1)\frac{1}{\sqrt{2}} + D(i,2)\sin(t) + D(i,3)\cos(t) + D(i,4)\sin(2t) + D(i,5)\cos(2t) + \cdots \tag{2}$$

and plotted over the range $-\pi < t < \pi$. Thus, each data point (e.g. network node) may be viewed as a curve between $-\pi$ and π. This function representation has several interesting characteristics

DOI: 10.1002/dac

since it preserves the standard deviation and the distances of data points (e.g. close points will appear as close curves while distant points as distant curves). Hence, if there is an underlying structure in the data, it may be visualized by their Andrews' curves. More specifically, by studying Andrews' curves in conjunction with clustering process, we can claim that the different shapes of curves among clusters are an indication of dissimilarity between nodes belonging to different clusters, whereas similar curves among nodes of the same cluster are an indication of similarity between them.

In our framework, Andrews' curves can graphically prove the fact that clustering succeeds in discovering groups of similar nodes on the basis of their requests. In particular, since each node can be represented by its demand vector $D(i, :)$, where $i = 1, \ldots, n$, it could be feasible for each node to be graphically depicted by a single curve based on the w variables: $D(i, :) = (D(i, 1), \ldots, D(i, w))$ which express the packets that the node n_i requests on each of the w channels. Therefore, we expect that the irregularities in the demand matrix, which are due to different nodes' packet requests, will be smoothened after clustering process that groups together nodes with similar requests.

For this part of experimentation we fixed the number of nodes at $n = 40$ while we set $w = 10$ channels. The traffic load was set to $k = \lfloor n * w/5 \rfloor$ following the model B, i.e. poisson traffic. We chose noc $= 3$ clusters to validate that our clustering algorithm succeeds in revealing the three classes of nodes, i.e. light, medium and heavy loads, which are described in Section 6. As depicted in Figure 4(a), where each node is represented by a single curve based on 10 variables ($w = 10$ channels), there are different nodes' curves within the demand matrix D implying nodes' dissimilarities. At the same time, the existence of curves' groups is apparent indicating similarities between some nodes. Thus, in Figure 4(b), we can note that after clustering process nodes' curves have strong similarity within individual clusters, whereas clusters' curves are separated and therefore well discriminated. As it is also expected, we have three different curve shapes because nodes were divided in advance into three classes. The number of curves in each subplot reveals the number of nodes on each cluster and thus C_1 contains 8 nodes, C_2 contains 20 nodes while C_3 contains 12 nodes. The fact that irregularities before clustering (Figure 4(a)) are smoothened after

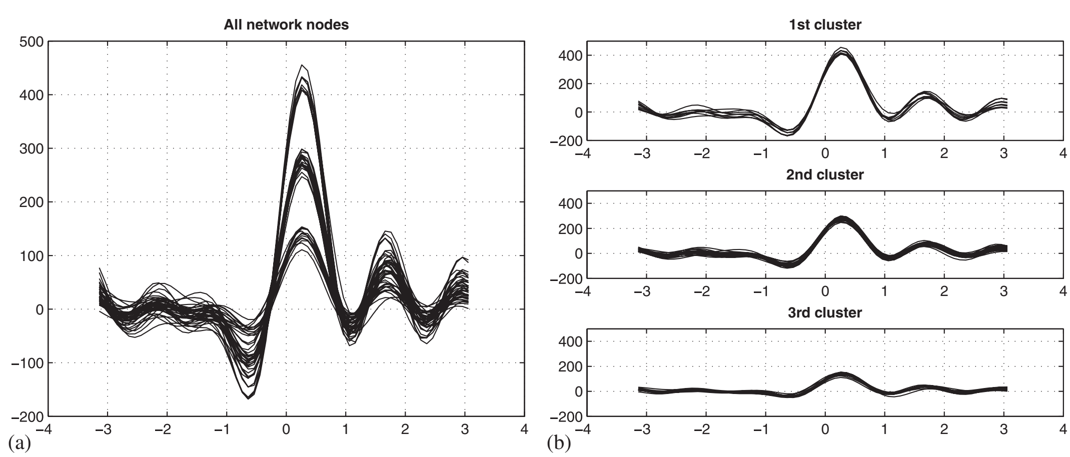


Figure 4. Andrews' curves: each curve depicts a node based on $w = 10$ channels: (a) nodes before clustering and (b) nodes after clustering.

DOI: 10.1002/dac

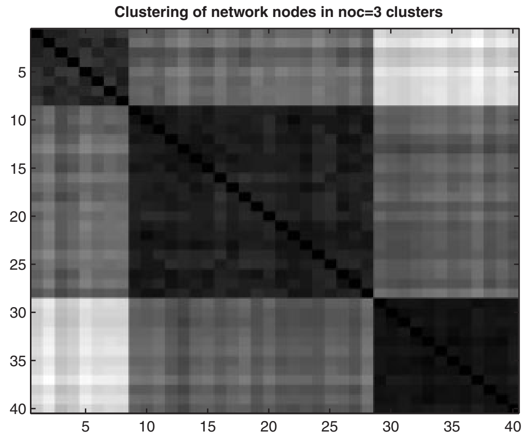


Figure 5. The clustering outline of $n=40$ nodes to noc $=3$ clusters.

clustering (Figure 4(b)) is a clear indication that K-means clustering algorithm creates clusters with members close to each other according to their packets requests.

Figure 5 provides an alternative way to evaluate the quality of the obtained clusters. In this figure we present the clustering outline under the above network parameters visualizing the similarities and dissimilarities between nodes' patterns in terms of the Euclidean distance measure. Particularly, given the $n\times w$ demand matrix D, we compute the $n\times n$ distance matrix E whose rows and columns have been rearranged so that nodes clustered together are put in consecutive rows (columns). In visualizing the distance matrix E, the darker the coloring of a cell (i, j), where $i, j=1,\ldots,n$, the more similar the nodes at positions i and j are. Thus, given that clusters contain the most similar nodes, the darker rectangles appear on the plot's diagonal and reveal the obtained clusters. Figure 5 depicts three clearly formed dark rectangles that correspond to the three clusters, while it also provides the clusters' membership on its x and y axes, i.e. C_1, C_2, C_3 contain $8, 20, 12$ nodes, respectively.

8. SIMULATION RESULTS

To evaluate the proposed algorithm, we carried out experimentation where we compare CBSA with POSA given that both schemes use the same prediction mechanism and employ the OIS as scheduling algorithm. We have conducted experiments with different number of nodes n, channels w and clusters noc, whereas we also evaluated the algorithms' performance under different traffic loads k, where k expresses the upper bound of nodes' requests per channel. We defined the line at 3 Gbps per channel, we considered the tuning time to be negligible, whereas the outcome results from 10 000 transmission frames. The performance of the compared algorithms is measured in terms of network throughput and mean packet delay.

Let us suppose that Γ denotes the network throughput while r represents the line transmission rate per channel in Gbps. Then, given that t represents the schedule's length and D the $n\times w$

DOI: 10.1002/dac

demand matrix, the network throughput is defined as follows:

$$\Gamma = \frac{\sum_{i=1}^{n} \sum_{j=1}^{w} d(i,j)}{t} * r \tag{3}$$

On the other hand, the mean packet delay is defined as the average time packets spend in the systems waiting to transmit and it is composed of packet transmission delay.

Figure 6(a) depicts the network's throughput as a function of the number of nodes for $n = 10, 20, \ldots, 100$, whereas the number of channels is set to $w = 5$ and the number of clusters is fixed at $\text{noc} = 6$. The traffic load follows the model A where the upper bound of nodes' requests is $k = \lfloor (n * w)/5 \rfloor$ for scalability reasons [14]. The throughput improvement in case of the CBSA scheme indicates that the use of the proposed algorithm leads to a significant reduction of the schedule's length. It is apparent that for any number of nodes n, CBSA provides steadily higher throughput compared with POSA. This is expected since, as it has already been discussed, CBSA prioritizes the long-length requests and thus allocates more free time slots for the rest requests.

Figure 6(b) presents the improvement on network's throughput when model B is employed. Not only is it obvious that our scheme achieves higher network throughput than POSA, but also CBSA exhibits better performance under asymmetric traffic. As discussed in Section 6 model B defines three classes of nodes, namely light, medium or heavy according to its traffic load, to simulate a more realistic environment. In this case, the clustering process of the CBSA manages to recognize and separate these classes and as a result it prioritizes clusters containing the heavy ones.

Mean packet delay is the second performance metric that is evaluated. For this part of experiments, we keep the same values of channels $w = 5$, traffic load $k = \lfloor (n * w)/5 \rfloor$ and clusters $\text{noc} = 6$ as previously, while we vary the number of nodes setting $n = 10, 20, \ldots, 100$. Figure 7(a) and (b) presents mean packet delay as a function of the network's throughput and validate the CBSA's superiority, since it is obvious that the improvement in network's throughput does not affect the mean packet delay. More specifically, Figure 7(a) depicts mean packet delay *versus* throughput in case of model A. It is clear that CBSA excels in network's throughput measurements and at the

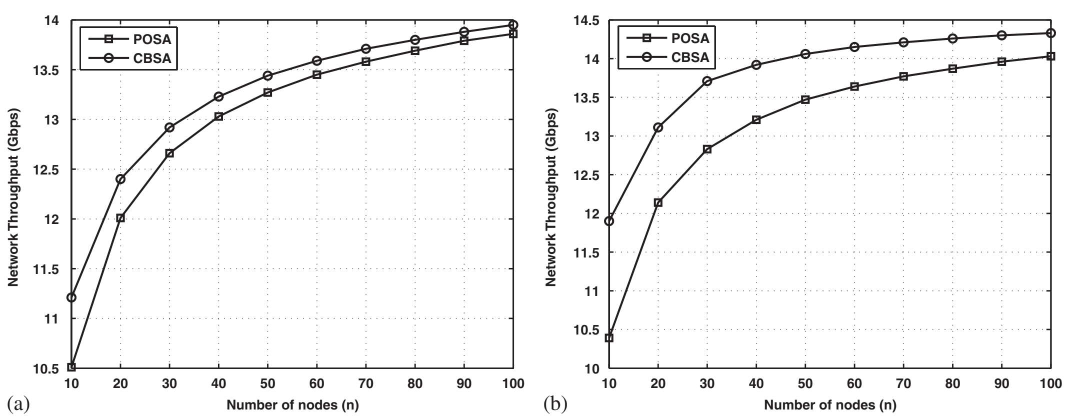


Figure 6. Network throughput as a function of the number of nodes for $w = 5$ channels, traffic load $k = \lfloor (n * w)/5 \rfloor$ and $\text{noc} = 6$ clusters: (a) uniform model and (b) asymmetric model.

DOI: 10.1002/dac

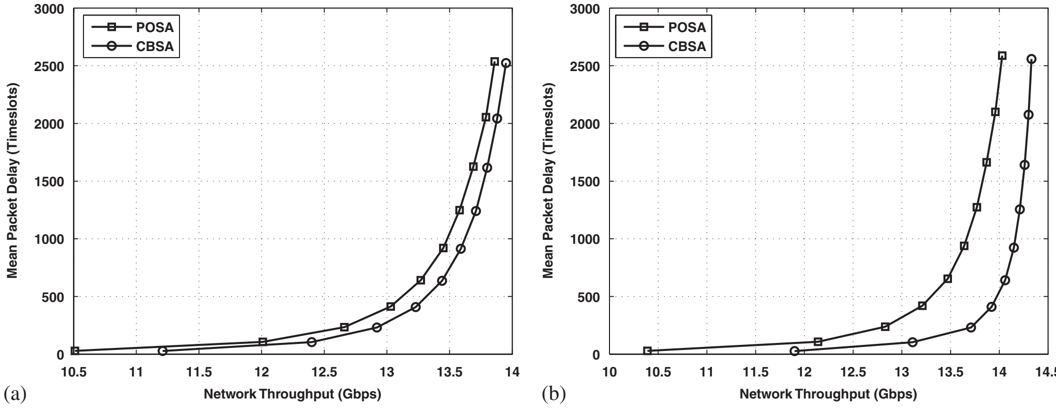


Figure 7. Mean packet delay as a function of the network throughput for $w=5$ channels, traffic load $k=\lfloor(n*w)/5\rfloor$ and noc$=6$ clusters: (a) uniform model and (b) asymmetric model.

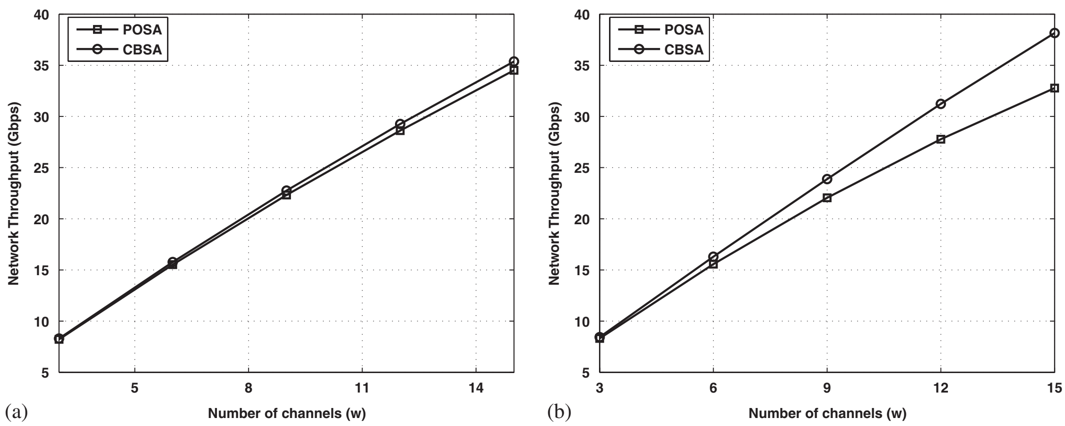


Figure 8. Network throughput as a function of the number of channels for $n=40$ nodes, traffic load $k=10$ and noc$=6$ clusters: (a) uniform model and (b) asymmetric model.

same time it causes less delay in comparison with POSA. The comparison betweens Figure 7(a) and (b) confirms the proposed scheme's superiority under asymmetric traffic in terms of both network's throughput and mean packet delay.

Given that the proposed scheme achieves high throughput levels, we evaluated network throughput under different number of channels and the results are illustrated in Figure 8(a) and (b). For this part of experimentation, we fixed the number of nodes at $n=40$, while we set the traffic load to $k=10$ and the number of clusters to noc$=6$. The number of channels are set to $w=3, 6, 9, 12$ and 15.

Figure 8(a) shows the network throughput *versus* the number of channels under uniform traffic, i.e. model A, whereas Figure 8(b) depicts the network throughput *versus* the number of channels under asymmetric traffic, i.e. model B. Based on these figures, we observe that in both algorithms the network throughput is increasing as the number of channels increases and this is natural, since the more the network channels the shorter the schedule length. In practice, there is more time space for nodes packets to be scheduled. It is remarkable, however, that CBSA is marginally better than POSA in the case of uniform traffic while it clearly outperforms POSA under asymmetric traffic. What is more, we note that CBSA's performance is getting better as the network channels increase.

In Figure 9(a) and (b) the network throughput is presented as a function of the traffic load under both models A and B. As it has already been discussed, the traffic load is expressed by the parameter k, which indicates the upper bound of the number of packets that a node destines to each channel. In this group of simulation, we carried out experiments with $k = 20, 30, 40, 50$ and 60 while we fixed the number of nodes at $n = 40$, the number of channels at $w = 5$ and the number of clusters at noc $= 6$. The proposed algorithm is presented to be steadily superior in comparison with POSA for both uniform and asymmetric traffics, while it is important to note that under asymmetric traffic its behavior is improved as the network load is increasing.

More specifically, Figure 9(a) shows that under uniform traffic both CBSA and POSA's performances are decreasing as the network load is increasing and this is expected, since when k increases it is difficult for the scheduling algorithm to find open space in the constructed scheduling matrix S. Consider, for example, that in the scheduling matrix S there is an open space of 10 slots and the next arriving request consists of 11 packets (uniform traffic), which means that it requires 11 time slots for its transmission. Given that CBSA and POSA should schedule the requested packets as a whole, they put the packets at the end of the matrix S. This leads to many unused time slots and, as a consequence, to decreased channel utilization as well as to decreased network throughput.

This marginal superiority, however, is due to the traffic model, which does not highlight the advantage of the proposed scheme. Figure 9(b) not only confirms our intuition that CBSA succeeds in scheduling asymmetric traffic better than POSA but also proves that the more asymmetric the network traffic, the better performance the CBSA can achieve. This is expected, since as

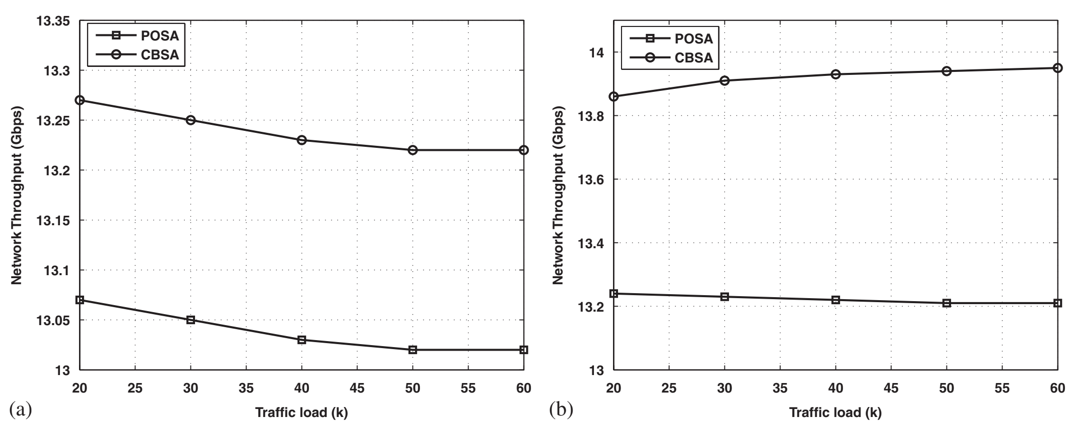


Figure 9. Network throughput as a function of the traffic load for $n = 40$ nodes, $w = 5$ channels and noc $= 6$ clusters: (a) uniform model and (b) asymmetric model.

DOI: 10.1002/dac

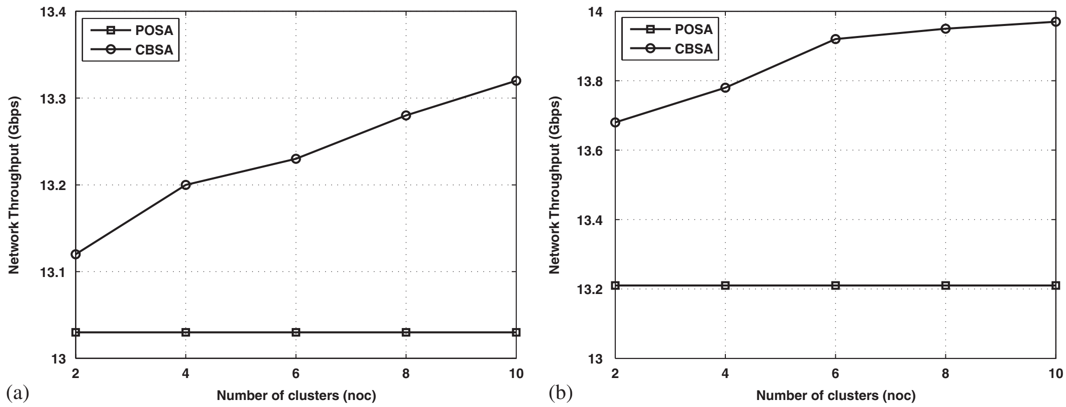


Figure 10. Network throughput as a function of the number of clusters for $n=40$ nodes, $w=5$ channels and traffic load $k=\lfloor (n*w)/5 \rfloor$: (a) uniform model and (b) asymmetric model.

the nodes traffic patterns become more dissimilar the clustering process of CBSA manages to separate them successfully and, thus, we take advantage of the obtained clusters. Consider, for example, the patterns $D(1,:)=(4,5,3)$ and $D(2,:)=(2,3,1)$ of users u_1 and u_2, respectively, which could be created under uniform traffic and the patterns $D(3,:)=(4,5,4)$ (heavy load node) and $D(4,:)=(1,2,0)$ (light load node) of users u_3 and u_4, respectively, which could be created under asymmetric traffic. CBSA would probably separate u_3 from u_4 more easily than u_1 from u_2. What is more, giving priority to u_3 instead of u_4 would take greater advantage than prioritizing u_1 instead of u_2.

Finally, given that the proposed algorithm aims at grouping together nodes with similar traffic patterns it was challenging to study its performance under different number of clusters. Thus, in the last plot of simulation we evaluated network throughput under different values of noc. In Figure 10(a) and (b) we fixed the number of nodes at $n=40$, the network load at $k=40$ while we set the number of channels to $w=5$. Under these parameters, we conducted experiments with noc $=2,4,6,8$ and 10.

What we can firstly comment based on Figure 10(a) and (b) is that, as expected, the performance of POSA is independent of the number of clusters, while CBSA outperforms POSA both under uniform (Figure 10(a)) and asymmetric traffic (Figure 10(b)). However, in accordance with all previous plots, it is clear that CBSA exhibits better performance under asymmetric than under uniform traffic and, thus, it clearly outperforms POSA when the network traffic is asymmetric. A second observation is that the CBSA's performance is increasing as the number of clusters is increasing and this is natural, since the obtained clusters become more coherent, which means that they contain more similar nodes. Of course, setting noc $=n$ would probably lead to better results. However, this means that we sort the nodes according to their requests instead of clustering them which implies higher computational complexity.

8.1. *Extended experimentation under real-time traffic*

On the above simulation results, all packets are considered to be of equal priority. However, since real-time traffic (high-priority packets) represents 25–30% of the Internet traffic, in this section, we have conducted experiments where the CBSA has been extended to handle prioritized traffic. During each frame, high- and low-priority packets are clustered and scheduled separately. Furthermore, the high-priority packets have the privilege of being scheduled prior to low-priority ones. Thus, the proposed scheme succeeds in obtaining significant improvements for real-time traffic, without sacrificing the performance for nonreal-time traffic.

More specifically, Figure 11(a) and (b) represents mean packet delay as a function of the network throughput in case of total traffic, high- and low-priority packets when the uniform and asymmetric models are employed, respectively. In these figures, the number of nodes is varied by setting $n=10, 20, \ldots, 100$, whereas the number of channels is fixed at $w=5$. The traffic load is set to $k=\lfloor (n*w)/5 \rfloor$ and noc is taken to be equal to six clusters. Based on the above figures, it is observed that the curves depicting the mean packet delay of POSA for high- and low-priority packets are very close. This is expected, since the POSA scheme does not take into account the packets' priority. On the other hand, under the CBSA scheme, the mean delay of high- and low-priority packets differs significantly. It is important to mention that CBSA clearly outperforms POSA in terms of mean delay of both high- and low-priority packets. Especially in the case of high-priority packets, the proposed CBSA scheme achieves a significantly lower delay in comparison with POSA, because such packets have the privilege of being scheduled prior to low-priority ones. For example, for $n=50$, the mean delay of high-priority packets in the case of CBSA is 76% lower than the corresponding mean delay of POSA for both uniform and asymmetric traffics. This significant improvement is not made in the cost of a high delay of low-priority packets, since as depicted in Figure 11(a) and (b) the low-priority packets' curves are very close to the total traffic curves.

The variance of delay is another important performance metric, especially in the case of real-time traffic [27]. In the presence of real-time traffic, it is crucial to keep the variance of delay of this traffic low to avoid long delays. Graphs, depicting the variance of delay for the POSA and CBSA schemes under prioritized real-time traffic, are given in Figure 12(a) and (b). More specifically, Figure 12(a) and (b) depicts the variance of delay of total traffic, high-priority and low-priority

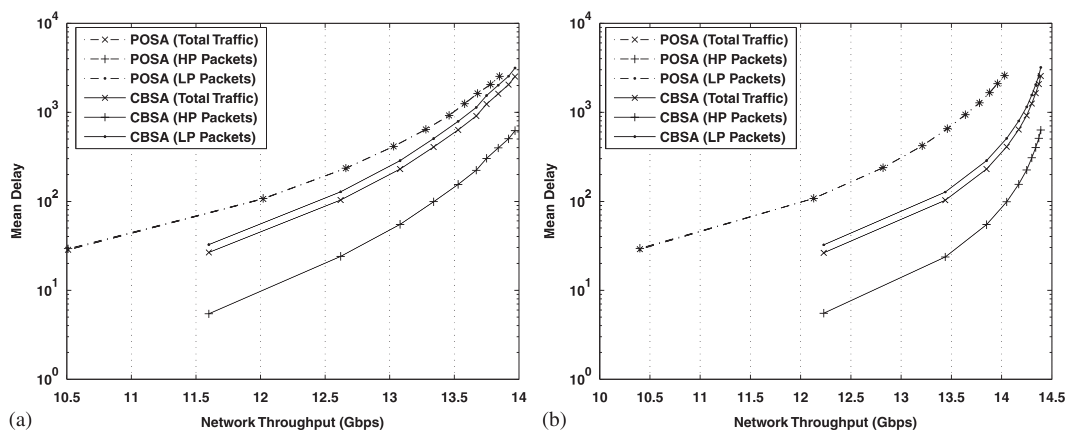


Figure 11. Mean packet delay of total traffic, high-priority and low-priority packets as a function of the network throughput: (a) uniform model and (b) asymmetric model.

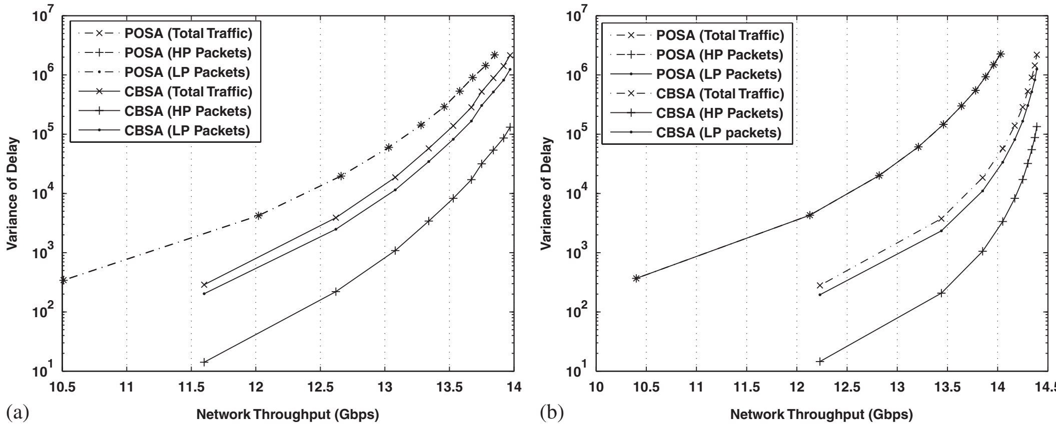


Figure 12. Variance of delay of total traffic, high-priority and low-priority packets as a function of the network throughput: (a) uniform model and (b) asymmetric model.

packets, as a function of the network throughput in the case of uniform and asymmetric model, respectively, for the above values on network's parameters (i.e. n, w, k and noc). Both figures indicate that CBSA significantly reduces the variance of delay for all type of packets and especially for high-priority ones. For example, when the CBSA scheme is used, for $n=50$, the variance of delay of high-priority packets is 94% lower than the one of POSA for both uniform and asymmetric traffics. Furthermore, it is notable that in the case of CBSA, the variance of delay of low-priority packets is lower than that of the total traffic and this is due to the fact that the proposed algorithm closely schedules packets of the same priority.

8.2. Major observations

The following conclusions can be extracted from the simulation results presented in Figures 6–12:

(1) For any number of nodes, channels and clusters as well as for any traffic model, the proposed CBSA scheme is superior to the POSA, since it creates a shorter schedule that advances the network's throughput and reduces the mean packet delay. This is due to the fact that CBSA takes into account the specific nodes' demands based on the clustering of the network's nodes.
(2) The superiority of CBSA over POSA is greater under the asymmetric traffic model than under the uniform one. This is due to the fact that asymmetric traffic significantly differentiates the network's nodes and, thus, the clustering process succeeds in creating more coherent clusters (i.e. clusters with similar nodes). What is more, the clusters themselves, in the case of asymmetric traffic, are more dissimilar and thus prioritizing those with heavy load nodes advances the scheduling process.
(3) The proposed scheme can be easily extended to handle real-time traffic (high-priority packets) such as audio or video. This extension provides a superior performance for high-priority packets in terms of mean delay and variance of delay, without sacrificing the performance of low-priority packets, such as text, e-mail or file transfer.

DOI: 10.1002/dac

9. CONCLUSIONS AND FUTURE WORK

A novel scheduling scheme that combines prediction and clustering techniques is introduced. The proposed CBSA scheme is based on the rearrangement of the network's nodes service order by organizing them into clusters according to their packet requests per channel. In this way, the individual traffic pattern of each node is taken into account and, consequently, the network performance is significantly improved.

The idea of using clustering algorithms for traffic scheduling is applicable to a broad range of networks, including optical and wireless LANs, and wireless push systems. We are currently working in this direction.

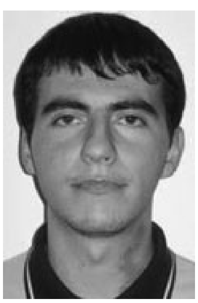
REFERENCES

1. Mukherjee B. *Optical WDM Networks*. Springer: Berlin, 2006.
2. Papadimitriou G, Obaidat M, Pomportsis AS. Advances in optical networking. *International Journal of Communication Systems* 2002; **15**(2–3):101–113.
3. Green P. Progress in optical networking. *IEEE Communications Magazine* 2001; **39**(1):54–61.
4. Papadimitriou G, Tsimoulas P, Obaidat M, Pomportsis A. *Multiwavelength Optical LANs*. Wiley: NY, 2003.
5. Wang B, Hou J, Han C. Design and analysis of a channel access protocol for wdm-based single-hop optical networks. *Photonic Network Communications* 2003; **6**(3):289–308.
6. Bernstein G, Rajagopalan B, Saha D. *Optical Network Control*: *Architecture*, *Protocols*, *and Standards*. Addison-Wesley: Reading, MA, 2003.
7. Pountourakis I, Stavdas A. A wdm control architecture and destination conflicts analysis. *International Journal of Communication Systems* 2002; **15**(2–3):161–174.
8. Chen M, Dono N, Ramaswami R. A media-access protocol for packet-switched wavelength division multiaccess metropolitan area networks. *IEEE Journal on Selected Areas in Communications* 1990; **8**(6):1048–1057.
9. Chlamtac I, Fumagalli A. Quadro-star: a high performance optical wdm star network. *IEEE Transactions on Communications* 1994; **42**(8):2582–2591.
10. Habbab I, Kavehrad M, Sundberg C. Protocols for very high-speed optical fiber local area networks using a passive star topology. *IEEE/OSA Journal of Lightwave Technology* 1987; **5**(12):1782–1794.
11. Mehravari N. Performance and protocol improvements for very high speed optical fiber local area networks using a passive star topology. *IEEE/OSA Journal of Lightwave Technology* 1990; **8**(4):520–530.
12. Sivalingam K, Wang J, Wu J, Mishra M. An interval-based scheduling algorithm for optical wdm star networks. *Photonic Network Communications* 2002; **4**(1):73–87.
13. Sivalingam K, Wang J. Media access protocols for wdm networks with on-line scheduling. *IEEE/OSA Journal of Lightwave Technology* 1996; **14**(6):1278–1286.
14. Johnson E, Sivalingam K, Mishra M. Scheduling in optical wdm networks using hidden Markov chain based traffic prediction. *Photonic Network Communications* 2001; **3**(3):269–283.
15. Sarigiannidis P, Papadimitriou G, Pomportsis A. A high throughput scheduling technique, with idle timeslot elimination mechanism. *IEEE Journal of Lightwave Technology* 2006; **24**(12):4811–4827.
16. Jain A, Murty M, Flynn P. Data clustering: a review. *ACM Computing Surveys* 1999; **31**(3):264–323.
17. Srivastava J, Cooley R, Deshpande M, Tan P-N. Web usage mining: discovery and applications of usage patterns from web data. *SIGKDD Explorations* 2000; **1**(2):12–23.
18. Petridou S, Koutsonikola V, Vakali A, Papadimitriou G. Time aware web users clustering. *IEEE Transactions on Knowledge and Data Engineering* 2008; DOI: 10.1109/TKDE.2007.190741.
19. Goldberger J, Gordon S, Greenspan H. Unsupervised image-set clustering using an information theoretic framework. *IEEE Transactions on Image Processing* 2006; **15**(2):449–458.
20. Kasturi J, Acharya R, Ramanathan M. An information theoretic approach for analyzing temporal patterns of gene expression. *Bioinformatics* 2003; **19**(4):449–458.
21. Albanese M, Picariello A, Sansone C, Sansone L. Web personalization based on static information and dynamic user behavior. *Proceedings of the 6th Annual ACM International Workshop on WIDM'04*, Washington, DC, U.S.A., November 2004; 80–87.

22. Bianco A, Mardente G, Mellia M, Munafo M, Muscariello L. Web user session characterization via clustering techniques. *GLOBECOM'05*. IEEE Press: New York, December 2005; 6.
23. Sivalingam K, Subramaniam S. *Emerging Optical Network Technologies: Architectures, Protocols and Performance*. Springer: Berlin, 2004.
24. Hastie T, Tibshirani R, Friedman J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer: Berlin, 2001.
25. Cleveland W. *Visualizing Data*. Hobart Press: Summit, NJ, U.S.A., 1993.
26. Andrews D. Plots of high-dimensional data. *Biometrics* 1972; **28**:125–136.
27. Diao J, Chu P. Packet rescheduling in wdm star networks with real-time service differentiation. *IEEE/OSA Journal of Lightwave Technology* 2001; **19**(12):1818–1828.

AUTHORS' BIOGRAPHIES

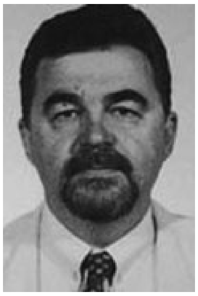
Sophia G. Petridou received the BS degree in Computer Science from the Aristotle University of Thessaloniki, Greece, in 2000, where she is currently working toward the PhD degree in Communication Networks. Her research interests include clustering, optical and wireless networks.



Panagiotis G. Sarigiannidis received the Diploma and PhD degrees in Computer Science from the Aristotle University of Thessaloniki, Greece, in 2001 and 2007, respectively. He is currently an adjunct lecturer at the University of Ioannina, Greece. His research interests include optical networks and optical switching.

Georgios I. Papadimitriou received the Diploma and PhD 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. In 1997 he joined the faculty of the Department of Informatics, Aristotle University of Thessaloniki, Greece, where he is currently serving as an Associate Professor. His main research interests include optical networks and wireless networks. Prof. Papadimitriou is Associate Editor of the IEEE Network, the IEEE Communications Magazine, the IEEE Transactions on Systems, Man and Cybernetics-Part C, the IEEE Transactions on Broadcasting, and the IEEE Sensors Journal. He is co-author of the books 'Optical Switching' (Wiley, 2007), 'Multiwavelength Optical LANs' (Wiley, 2003), and 'Wireless Networks' (Wiley, 2003), and co-editor of the book 'Applied System Simulation' (Kluwer, 2003). He is author or co-author of 150 journal and conference papers. He is a Senior Member of the IEEE.

DOI: 10.1002/dac



Andreas S. Pomportsis received the BS degree in Physics and the MS 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 PhD 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 'Optical Switching' (Wiley, 2007) and 'Multi-wavelength Optical LANs' (Wiley, 2003) and 'Optical Networks' (Wiley, 2003). His research interests include computer networks, computer architecture, parallel and distributed computer systems, and multimedia systems.

DOI: 10.1002/dac