A High Performance Clustering-Driven MAC Protocol for Single-Hop Lightwave Networks

S. G. Petridou, *Member, IEEE,* P. G. Sarigiannidis, *Member, IEEE,* G. I. Papadimitriou, *Senior Member, IEEE,* and A. S. Pomportsis

Abstract— **A Medium Access Control (MAC) protocol for Wavelength Division Multiplexing (WDM) star networks has to handle two key issues: channel assignment and nodes' service order. Traditional scheduling techniques define the nodes transmissions and receptions, by considering either channel assignment or nodes' service order issue. Furthermore, they take into account data channels or receivers' availability, but they ignore the similarity between nodes' demands which affect the scheduling length. In this paper a novel, clustering-driven scheduling approach is introduced. The proposed Clustering Driven - Minimum Scheduling Latency (CD-MSL) scheme combines all the aforementioned information and creates groups of source nodes which present similar demands on the basis of their message destination nodes. Based on the obtained clusters, CD-MSL improves network performance by rearranging the nodes' service order. Simulation results indicate that the proposed clustering-driven scheme leads to a significantly higher throughput-delay performance, in comparison to conventional scheduling algorithms.**

Index Terms— **Clustering, scheduling, WDM star networks.**

I. Introduction

THERE are two constraints on scheduling a message in WDM star networks: the data channels' availability as well as the receivers' availability. Thus, channel assignment and nodes' service order are two key issues in designing MAC protocols for optical WDM star networks [1]. Up to now, popular scheduling techniques consider either channel assignment or nodes' service order issue but not both of them, and, thus, they suffer from low performance, especially when operating under heavy traffic.

A well-known, efficient scheduling algorithm for local area WDM star networks with broadcast-and-select architecture is the Earliest Available Time Scheduling (EATS) [2]. EATS addresses the channel assignment without, however, handling nodes' service order, since it considers source nodes in a sequential order, ignoring the fact that rearranging the nodes' service order may affect the network's performance. For example, if a message is destined to a busy node, its transmission time will be scheduled far later than the data channel's earliest available time, downgrading the channel utilization. The Receiver Oriented-Earliest Available Time Scheduling (RO-EATS) [3] is an extension of the EATS which overcomes the aforementioned drawback by taking into account the receiver's availability. The RO-EATS's core idea is to prioritize messages that are destined for the least used receiver, while it retains the EATS's logic for the channel assignment issue. As a result, RO-EATS is significantly improved in comparison to EATS in terms of mean packet delay without, however, providing remarkable improvements in terms of network throughput. A different approach for improving the network's performance is adopted by the Minimum Scheduling Latency (MSL) [4]. MSL retains the EATS's logic for the nodes' service, while it modifies the choice of the transmission data channel on the basis of the minimum scheduling latency i.e. the channel for which the shorter time interval will pass from the time it becomes available. In practice, it deals with the channel assignment issue taking into account not only the channel's availability but also the receiver's availability. Even though MSL clearly outperforms EATS and RO-EATS, its performance is limited by the fact that it retains the EATS's logic for the nodes' service order.

This paper introduces a novel algorithm that deals with both channel assignment and nodes' service order issues, based on the clustering [5], [6] of the network's nodes. More specifically, the proposed Clustering Driven - Minimum Scheduling Latency (CD-MSL) handles the channel assignment issue according to the MSL logic which takes into account both channels' and receivers' availability information. At the same time, it addresses the nodes' service order in an innovative way which combines information considering both the source and destination nodes without imposing complexity to the scheduling algorithm.

In general, clustering aims at creating groups of objects (i.e. clusters) on the basis that objects assigned to the same cluster are "similar" to each other and "dissimilar" to the nodes belonging to other clusters [5]. In our framework, CD-MSL groups together nodes with common message destination and, then, it defines the nodes' service order choosing for transmission nodes belonging to different clusters. Rearranging the nodes' service order in this way, CD-MSL manages to decrease the probability of scheduling messages to the same destination at successive order which downgrades the channel utilization. As a result, the schedule length is reduced and the network performance is upgraded. At the same time, clustering runs in time linear with the number of nodes without aggravating scheduling algorithm's complexity.

The remainder of this paper is organized as follows. Sec-

S. G. Petridou is with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: spetrido@csd.auth.gr)

P. G. Sarigiannidis is with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: sarpan@csd.auth.gr)

G. I. Papadimitriou is with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: gp@csd.auth.gr)

A. S. Pomportsis is with the Department of Informatics, Aristotle University, 54124 Thessaloniki, Greece (e-mail: apombo@csd.auth.gr)

TABLE I
NETWORK SYMBOLS' NOTATION

Symbol	Description
n, w	Number of nodes and data channels
$S = \{s_1, \ldots, s_n\}$	The set of source nodes
$D = \{d_1, \ldots, d_n\}$	The set of destination nodes
$\Lambda = \{\lambda_1, \ldots, \lambda_w\}$	The set of data channels
λ_0	The control channel
M	The $n \times n$ message table
k	The upper bound of messages' length
t	Schedule's length in timeslots
$L = \{l_1, \ldots, l_t\}$	The set of timeslots
H	The $w \times t$ scheduling matrix

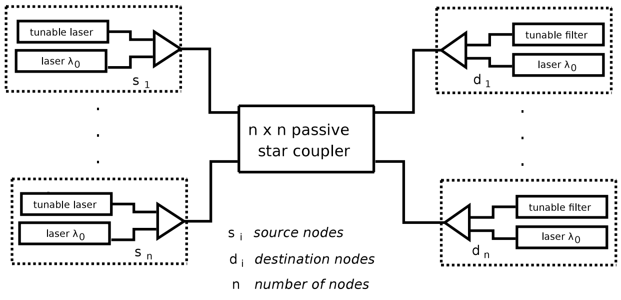


Fig. 1. The network architecture

tion II provides the network architecture, while clustering background is given in Section III. The proposed scheduling algorithm is described in Section IV, while Section V discusses the simulation results. Conclusions and future work insights are given in Section VI.

II. Network Architecture

Consider a single-hop, broadcast-and-select WDM star network consisting of n nodes and $w+1$ channels (wavelengths), where $\Lambda = \{\lambda_1, \cdots, \lambda_w\}$ is the set of the data channels, while one channel λ_0 is used for coordination (i.e. control channel). Each node is connected to a passive star coupler via a pair of optical fibers, one of which is used for transmission and the other for reception. Given that each of the n nodes can either transmit or receive a message, we use the sets $S = \{s_1, \ldots, s_n\}$ and $D = \{d_1, \ldots, d_n\}$ to denote the role of a node as a source or a destination, respectively. Transmissions from all nodes to all channels are combined at the passive star coupler and broadcast to all nodes via receiver fibers. More specifically, each node is equipped with two transmitters and two receivers. The first transmitter and receiver are fixed and tuned to λ_0 (FT-FR) for transmitting or receiving the control packets, while the second transmitter and receiver are tunable (TT-TR) for accessing the data channels. In this CC-FTTT-FRTR implementation, which is depicted in Fig. 1, two nodes s_i and d_j, $i, j = 1, \ldots, n$ and $i \neq j$, are able to communicate when the transmitter of s_i and the receiver of d_j are tuned to the same wavelength.

In the above CC-FTTT-FRTR system, each transmission frame is divided into two phases, namely the control and data phase. During the control phase, the control channel λ_0 is shared using the TDMA technique to avoid collisions of control packets [2]. In channel λ_0, each frame consists of n timeslots, where each timeslot is dedicated to one source node i.e. the node s_i, $i = 1, \ldots, n$. For example, the node s_i uses the i-th timeslot during the control phase to transmit its control packet. During the data phase, the real message transmission takes place. The nodes are assumed to generate messages of variable lengths which can be divided into several equal-sized packets. Each packet is transmitted in time equal to a timeslot. Thus, the data channels are also time-slotted, but the length of a timeslot is independent of the length of control frame [2]. We define $L = \{l_1, \ldots, l_t\}$ to be the set of the t timeslots which specify the schedule length of the data phase.

In such a network, given the tunability of the transmitters, it is obvious that two or more source nodes might cause channel collision, transmitting messages on the same data channel simultaneously. Furthermore, receiver collisions might occur when two or more nodes transmit messages to the same node simultaneously, since each node is equipped with a single tunable receiver [7]. Thus, the media access control (MAC) protocol has to coordinate the nodes' data transmission and prevent collisions [8], [9]. In our framework, the MAC protocol handles the above issues running a scheduling algorithm at the end of the control phase in each frame. This algorithm uses two tables, namely the Receiver Available Time (*RAT*) and the Channel Available Time (*CAT*) tables [2], [3], [4].

The *RAT* table consists of n elements, where $\mathit{RAT}(d_i) = x$, $i = 1, \ldots, n$, denotes that destination node d_i will be available after x timeslots. The *CAT* table contains w elements, where $\mathit{CAT}(i) = y$, $i = 1, \ldots, w$, indicates that channel i will be available after y timeslots. Tables *RAT* and *CAT* are used to avoid receiver and channel collisions, respectively. Based on these tables, the scheduling algorithm produces a $w \times t$ scheduling matrix H, where t denotes the length of the schedule in timeslots. Each element $h(i, j)$, $i = 1, \ldots, w$ and $j = 1, \ldots, t$, represents the destination node that receives a message on channel λ_i, during the timeslot l_j.

III. Clustering Background

Clustering is an important step in the process of data analysis with applications to numerous fields such as [6], [10]. Informally, clustering is defined as the problem of partitioning data objects into groups (i.e. clusters), such that objects in the same group are "similar", while objects in different groups are "dissimilar".

In our framework, the clustering process aims at partitioning the source nodes of set S. The source nodes are considered to be similar and, thus, they are grouped together if they present common message destination. Therefore, we organize the sets S and D into an $n \times n$ message table M, whose $m(i, j)$ element, $i, j = 1, \ldots, n$ and $i \neq j$, indicates the length of the message from the source node s_i to the destination node d_j. Given that each node s_i can transmit a message per frame, it is obvious that the i-th row of the M table will have one non-zero value. On the other hand, the j-th column of the M table can have more than one non-zero values indicating that each d_j node can receive more than one messages during a frame. Under this notation, each node s_i is considered to be a

TABLE II
CLUSTERING SYMBOLS' NOTATION

Symbol	Description
Cl	Clustering process
noc	Number of clusters
C_j	Cluster, $j = 1, \ldots, noc$
c_j	Cluster representative
Means	The $noc \times n$ clusters representatives' message table
d_E	Nodes distance over their messages' destination
J	Objective function

multivariate vector consisting of n values. We call this vector as demand pattern and we define it as follows:

$$M(i,:) = (m(i,1), \ldots, m(i,n))$$

A clustering Cl of S is a partition of S into noc disjoints sets i.e. clusters $C_1, \ldots, C_{noc}$, that is, $\bigcup_{i=1}^{noc} C_i = S$ and $C_i \bigcap C_j = \emptyset$ for all $i \neq j$. The noc clusters $C_1, \ldots, C_{noc}$ consist of $|C_1|, \ldots, |C_{noc}|$ members (i.e. source 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 the destination of their messages.

The clustering definition assumes that there is a quality measure that captures intra-cluster similarity and/or inter-cluster dissimilarity, and then clustering becomes the problem of grouping together data objects so that the quality measure is optimized. A common approach is to evaluate the dissimilarity between two objects (in our case the source nodes) by using a distance measure [5]. In our case, we proceed to the clustering of S using the Squared Euclidean distance, which is a well-known and widely used distance measure in the vector-space model [5], [6]. Therefore, the dissimilarity between two source nodes e.g. $s_x, s_y \in S$ can be evaluated by the distance of their vectors. Thus, we use the expression $d_E(s_x, s_y)$ to denote the Squared Euclidean distance of the nodes' vectors $M(x,:)$ and $M(y,:)$:

$$d_E(s_x, s_y) = \|M(x,:) - M(y,:)\|^2$$

Once the clusters are obtained, we consider an arbitrary cluster C_j, $j = 1, \cdots, noc$, of the set S. The representation of 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 which does not correspond to an existing node). This point is called cluster's representative c_j (also known as centroid), since each node $s_i \in C_j$ is represented by c_j. Given the vectors of $s_i \in C_j$, the vector of c_j is defined as follows:

$$\mathit{Means}(j,:) = \frac{1}{|C_j|} \sum_{s_i \in C_j} M(i,:), j = 1, \ldots, noc$$

Since both $M(i,:)$ and $\mathit{Means}(j,:)$ are vectors, their dissimilarity is measured by their Squared Euclidean distance $d_E(s_i, c_j)$. Considering all clusters, the clustering process is guided by the *objective function* J which is defined to be the sum of distances between each source node and the representative of the cluster that the node is assigned to:

$$J = \sum_{j=1}^{noc} \sum_{s_i \in C_j} d_E(s_i, c_j)$$

A clustering Cl that optimizes (minimizes) J groups together nodes from the set S that probably have common destination, i.e. nodes from the set D.

IV. The proposed Clustering-Driven Scheduling

The proposed CD-MSL scheme is a two-step approach which firstly defines the nodes' service order driven by the clustering process and then, it deals with the channel assignment issue following the MSL spirit. The core idea is that both the source and destination nodes should be taken into account in determining the nodes' service order. Thus, the proposed algorithm groups together source nodes with the same message destination. The goal is that messages destined for the same destination should not be scheduled in a successive order. Therefore, CD-MSL schedules in sequence messages from nodes belonging to different clusters. Furthermore, CD-MSL prioritizes clusters as well as the members of each cluster according to the length of their messages.

More specifically, during the first step, i.e. the clustering step, we employ the K-means, a widely used partitional clustering algorithm [11], in order to produce the Cl clustering of S. Then, given the Cl and the $n \times n$ message table M, we sort the members of each cluster according to the length of their messages and the result is recorded on *SortedM*. Similarly, using the *Means* table, consisting of the clusters representatives' vectors *Means*$(j,:)$, we compute the *SortedC* which contains the sorted clusters. The calculated *SortedM* and *SortedC* are then used in order to define the nodes' service order (*NSO*). Once the *NSO* is formed, the algorithm proceeds to the second step, called the channel assignment step. The goal of the function *ChannelAssignment* is to address the channel assignment issue, providing collision-free communication. Thus, it builds the scheduling matrix H based on the MSL algorithm's logic, which considers both *RAT* (i.e. receivers' availability) and *CAT* table (i.e. channels' availability).

Algorithm 1 The CD-MSL flow control

```
Input: A set $S$ of $n$ nodes organized in an $n \times n$ message
    table $M$, the upper bound on nodes' requests $k$ and the
    number of clusters $noc$.
Output: The scheduling matrix $H$.
    /*Clustering Step*/
 1: $(Cl, Means) = K - means(M, noc)$
 2: $SortedM = \mathit{Quicksort}(M, Cl)$
 3: $SortedC = \mathit{Quicksort}(Means)$
 4: $NSO = \mathit{ServiceOrder}(SortedM, SortedC)$
    /*Channel Assignment Step*/
 5: $H = \mathit{ChannelAssignment}(NSO)$
```

The CD-MSL has time complexity $O(nw^2)$ which is equal to the complexity of MSL. For the sake of brevity the proof is omitted.

To facilitate the comprehension of the proposed scheme, let us consider a WDM star network consisting of the source nodes $(s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8)$, the data channels $(\lambda_1, \lambda_2, \lambda_3)$ and having the upper bound of nodes' messages length $k = 6$ packets. Then, a 8×8 message table M could be the following:

$$M = \begin{pmatrix} 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 2 & 0 & 0 & 0 \\ 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 3 & 0 & 0 & 0 \end{pmatrix}$$

Example 1: In the above message table M the fact that $M(4,3) = 6$ means that the source node s_4 sends a message of length 6 to the destination node d_3. $\square$

TABLE III

THE TABLE *Members* BEFORE MEMBERS' SORTING

C_1	s_4				
C_2	s_1	s_2	s_3	s_7	s_8
C_3	s_5	s_6			

Applying the K-means algorithm for $noc = 3$ in the above message table M results in $Cl = (2,2,2,1,3,3,2,2)$ which can be represented by the Table III i.e. the table *Members* before member's sorting. From Table III, it holds that the node $s_4 \in C_1$, the nodes $s_1, s_2, s_3, s_7, s_8 \in C_2$, while the nodes $s_5, s_6 \in C_3$. As discussed in Section III, Cl places in the same cluster, source nodes which are similar in terms of their destination nodes, e.g. the nodes s_1, s_3 and s_8 destine their messages for node d_5, while the nodes s_5 and s_6 for node d_7. Node s_4 constitutes a cluster, i.e. C_1, by its own, since it has the most dissimilar pattern among all nodes. Finally, the rest nodes, i.e. s_2 and s_7, are forced to be placed in cluster C_2 since their patterns are more similar with that of the nodes belonging to C_2.

Sorting the members on each cluster according to the length of their message results in swapping the nodes of C_2. Therefore, the Table III is updated as follows:

TABLE IV

THE TABLE *Members* AFTER MEMBERS' SORTING

C_1	s_4				
C_2	s_8	s_3	s_1	s_2	s_7
C_3	s_5	s_6			

Given the above Cl clustering that K-means algorithm produces, the clusters representatives' message table *Means* is formed according to the vectors of clusters' members:

$$\mathit{Means} = \begin{pmatrix} 0 & 0 & 6 & 0 & 0 & 0 & 0 & 0 \\ 0.2 & 0 & 0 & 0 & 1.2 & 0.2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 3.5 & 0 \end{pmatrix}$$

Sorting *Means*, according to the length of clusters' representatives, provides our algorithm with the following service order: C_1, C_3, C_2. At this point, given that each cluster consists of nodes with probably the same destination, our scheme should separate them taking into account the result of the *Means* sorting. Therefore, the nodes' service order is defined as $s_4, s_5, s_8, s_6, s_3, s_1, s_2, s_7$ instead of the sequential one $s_1, s_2, s_3, s_4, s_5, s_6, s_7, s_8$.

TABLE V

THE SCHEDULING MATRIX H PRODUCED BY CD-MSL

	Timeslots							
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8
λ_1	d_3	d_3	d_3	d_3	d_3	d_3	d_7	d_7
λ_2	d_7	d_7	d_7	d_7	d_7	d_1	d_6	
λ_3	d_5	d_5	d_5		d_5	d_5		d_5

TABLE VI

THE SCHEDULING MATRIX H PRODUCED BY EATS

	Timeslots									
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9	l_{10}
λ_1	d_5	d_3	d_3	d_3	d_3	d_3	d_3	d_5	d_5	d_5
λ_2	d_1	d_7	d_7	d_7	d_7	d_7	d_6			
λ_3			d_5	d_5				d_7	d_7	

Table V depicts the scheduling matrix H produced by the proposed CD-MSL algorithm when the transmitters/receivers tuning time is set to 1 and the propagation delay of messages is set to 2. On the other hand, Tables VI, VII and VIII represent the scheduling matrix H, in case that the EATS, RO-EATS and MSL algorithms are employed, respectively. Based on the above tables, the channel utilization providing by the CD-MSL is 87.5%, which is significantly improved in comparison with EATS, whose channel utilization is 70%, as well as, with that of RO-EATS and MSL, which both provide 77.8% utilization. In terms of the mean packet delay, the observed values are 3.2 (CD-MSL), 4.1 (EATS), 3.3 (RO-EATS) and 3.4 (MSL) timeslots. Thus, the proposed CD-MSL scheme clearly outperforms the EATS, RO-EATS and MSL approaches.

V. EXPERIMENTATION

To evaluate the proposed algorithm we carried out experiments, where we compare CD-MSL with EATS, RO-EATS and MSL. We experimented with different network parameters, however, due to the lack of space we present a small portion of the results, where we vary the number of nodes n and the traffic load k, where k expresses the upper bound of message

TABLE VII

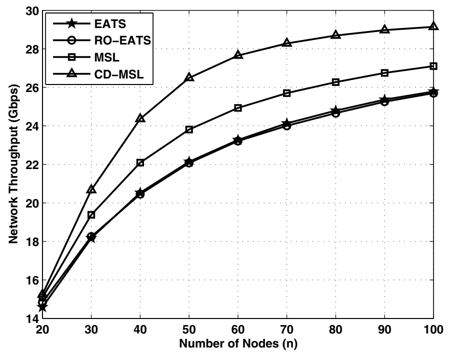
THE SCHEDULING MATRIX H PRODUCED BY RO-EATS

	Timeslots								
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
λ_1	d_1	d_6	d_5	d_5		d_5	d_5	d_5	
λ_2	d_3	d_3	d_3	d_3	d_3	d_3		d_7	d_7
λ_3	d_5	d_7	d_7	d_7	d_7	d_7			

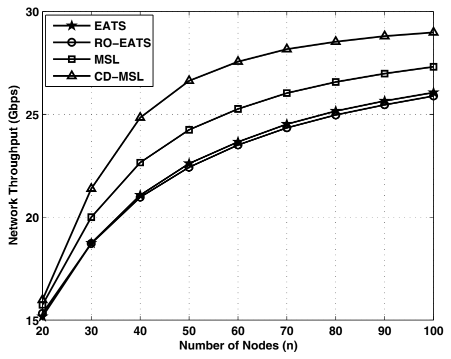
TABLE VIII

THE SCHEDULING MATRIX H PRODUCED BY MSL

	Timeslots								
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
λ_1	d_5		d_5	d_5	d_6	d_5	d_5	d_5	
λ_2	d_1	d_7	d_7	d_7	d_7	d_7		d_7	d_7
λ_3	d_3	d_3	d_3	d_3	d_3	d_3			

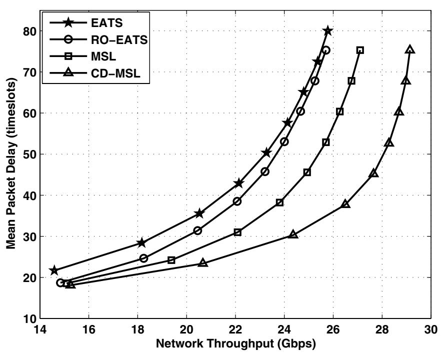


(a) Uniform traffic

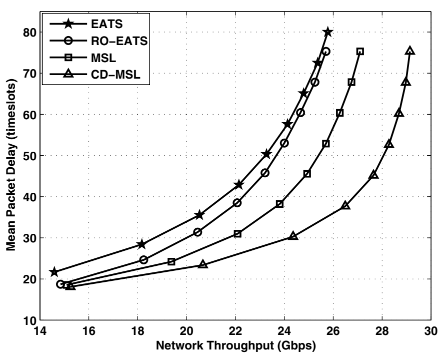


(b) Poisson traffic

Fig. 2. Network throughput as a function of the number of nodes for $w = 10$ channels, traffic load $k = 30$ and $noc = 10$ clusters.



(a) Uniform traffic



(b) Poisson traffic

Fig. 3. Mean packet delay as a function of the network throughput for $w = 10$ channels, traffic load $k = 30$ and $noc = 10$ clusters.

length requested per node per frame. The performance of the compared algorithms is evaluated in terms of network throughput and mean packet delay.

Traffic modeling is based on two distinct models. According to the first model, it is assumed that the packet arrival process at each of the queues follows the Uniform distribution, while according to the second model, the packet arrival process is assumed that it follows the Poisson distribution $p(X;\theta) = e^{-\theta}\lambda^X/X!$. The second model proceeds to the generation of nodes load patterns defining three classes of nodes: light, medium or heavy according to their traffic load. The values of θ for these three classes are defined as $k/4$, $k/2$ and $3k/4$ [12], respectively, while each node is assigned to a class with equal probability.

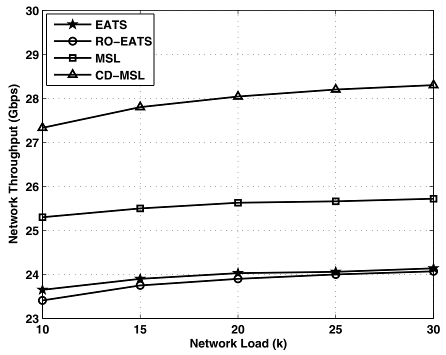
The simulation results are produced according to the following assumptions:

1) The transmitters/receivers tuning time is set to 1 and the propagation delay of messages is set to 2.
2) The line transmission rate per channel is set to 3 Gbps.
3) The outcome results from 10000 transmission frames.

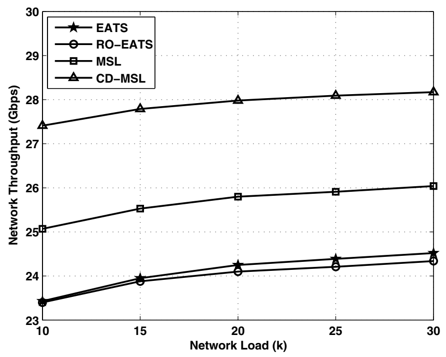
A. Simulation Results

Fig. 2(a) and 2(b) depict the network's throughput as a function of the number of nodes for $n = 20, 30, \ldots, 100$, while the traffic load is set to $k = 30$, following the uniform and poisson traffic, respectively. The number of channels is set to $w = 10$, while the number of clusters is also defined at $noc = 10$. Defining $noc = w$, the nodes' service order is formed by choosing source nodes belonging to different clusters. In this way, consecutive messages are not scheduled to the same destination, since these messages probably have different destinations. The throughput improvement in case of the CD-MSL algorithm denotes that its schedule length is reduced. It is apparent that for any number of nodes n, the proposed CD-MSL scheme provides steadily higher throughput compared to EATS, RO-EATS and MSL under both uniform and poisson traffic. In particular, the minimum observed differences occur for $n = 20$ nodes, where the contribution of the clustering is of low value, since on the average each cluster has two members. Remarkable improvements are observed for $n > 40$ nodes, since the ratio between noc and n significantly contributes to the performance of CD-MSL.

In Fig. 3(a) and 3(b), which represent the mean packet delay as a function of the network's throughput under uniform and poisson traffic respectively, we notice that the improvement in network's throughput does not affect the mean packet delay. According to these figures, it is clear that the CD-MSL keeps the mean packet delay lower in comparison to the other protocols independently of the number of nodes, while it obtains a higher throughput. Indicatively, for $n = 50$ under uniform traffic, CD-MSL achieves a throughput of 26.5 Gbps and a mean packet delay of 37.7 timeslots, while the respective values for EATS are 22.0 Gbps and 42.9 timeslots, for RO-EATS 22.1 Gbps and 38.5 timeslots and for MSL 23.8 Gbps



(a) Uniform traffic



(b) Poisson traffic

Fig. 4. Network throughput as a function of the traffic load for $n = 70$ nodes, $w = 10$ channels and $noc = 10$ clusters.

and 38.2 timeslots.

Finally, we evaluate the network performance under different loads. More specifically, in this group of simulation, we experimented with $k = 10, 15, \ldots, 30$, while we fixed the number of nodes at $n = 70$, the number of channels at $w = 10$ and the number of clusters at $noc = 10$. Fig. 4(a) and 4(b) use the uniform and poisson traffic model respectively and denote that the proposed CD-MSL scheme is significantly superior in comparison to EATS, RO-EATS and MSL.

VI. Conclusions and Future Work

This paper introduces and evaluates a novel scheduling scheme which is driven by clustering techniques. The proposed CD-MSL approach handles the channel assignment issue taking into account both channels and receivers' availability, while it addresses the nodes' service order issue in an innovative clustering-driven way, which considers both the source and destination nodes. Taking into account the individual traffic pattern of each source node and prioritizing them properly, the CD-MSL manages to significantly upgrade the network performance.

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

References

[1] B. Mukherjee, *Optical WDM Networks*. Springer, 2006.

[2] F. Jia, B. Mukherjee, and J. Iness, "Scheduling variable-length messages in a single-hop multichannel local lightwave network," *IEEE/ACM Transactions on Networking*, vol. 3, no. 4, pp. 477–488, 1995.

[3] M. Ma, B. Hamidzadeh, and M. Hamdi, "An efficient message scheduling algorithm for wdm lightwave networks," *Computer Networks*, vol. 31, no. 20, pp. 2139–2152, 1999.

[4] J. Diao and P. Chu, "Packet rescheduling in wdm star networks with real-time service differentiation," *IEEE/OSA Journal of Lightwave Technology*, vol. 19, no. 12, pp. 1818 – 1828, 2001.

[5] A. K. Jain, M. N. Murty, and P. J. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, no. 3, pp. 264–323, 1999.

[6] S. Petridou, V. Koutsonikola, A. Vakali, and G. Papadimitriou, "Time aware web users clustering," *IEEE Transactions on Knowledge and Data Engineering*, to appear, 2008.

[7] G. Papadimitriou, P. Tsimoulas, M. Obaidat, and A. Pomportsis, *Multiwavelength Optical LANs*. NY: Wiley, 2003.

[8] P. Sarigiannidis, G. Papadimitriou, and A. Pomportsis, "A high throughput scheduling technique, with idle timeslot elimination mechanism," *IEEE/OSA Journal of Lightwave Technology*, vol. 24, no. 12, pp. 4811–4827, 2006.

[9] G. Papadimitriou, C. Papazoglou, and A. Pomportsis, "Optical switching: switch fabrics, techniques, and architectures," *IEEE/OSA Journal of Lightwave Technology*, vol. 21, no. 2, pp. 384 – 405, 2003.

[10] J. Srivastava, R. Cooley, M. Deshpande, and P.-N. Tan, "Web usage mining: Discovery and applications of usage patterns from web data," *SIGKDD Explorations*, vol. 1, no. 2, pp. 12–23, 2000.

[11] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.

[12] E. Johnson, M. Mishra, and K. Sivalingam, "Scheduling in optical wdm networks using hidden markov chain based traffic prediction," *J. Photonic Network Commun.*, vol. 3, no. 3, pp. 271–286, 2001.

Sophia G. Petridou received the B.S. degree in Computer Science from the Aristotle University of Thessaloniki, Greece in 2000, where she is currently working toward the Ph.D. degree in communication networks. Her research interests include clustering, optical and wireless networks.

Panagiotis G. Sarigiannidis ($S'05$, $M'07$) received the Diploma and Ph.D. 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 ($M'89$, $SM'02$) received the Diploma and Ph.D. degrees in Computer Engineering from the University of Patras, Greece in 1989 and 1994 respectively. 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 five IEEE journals. He is co-author of three international books published by Wiley. He is author or coauthor of 70 journal and 80 conference papers. He is a Senior Member of the IEEE.

Andreas S. Pomportsis received the Diploma degree in electrical engineering, the M.S. degree in electronics and communications and the Ph.D. degree in Computer Science, all from the Aristotle University of Thessaloniki, Greece. Currently, he is a Professor with Department of Informatics, Aristotle University of Thessaloniki. He is co-author of three international books published by Wiley. He has published over 170 journal and conference papers. His research interests include computer networks and distributed systems.