An Efficient Clustering Oriented Algorithm for Message Scheduling on WDM Star Networks

Sophia G. Petridou, Panagiotis G. Sarigiannidis, *Graduate Member, IEEE*
Georgios I. Papadimitriou, *Senior Member, IEEE*, Andreas S. Pomportsis
Aristotle University
email:{spetrido, sarpan, gp, apombo}@csd.auth.gr

***Abstract*— Message sequencing and channel assignment are two important issues that have to be addressed when designing MAC protocols for optical Wavelength Division Multiplexing (WDM) networks. Up to now, popular approaches deal with channel assignment without however addressing the order in which the messages are scheduled. This paper presents a new reservation-based message scheduling algorithm for WDM star networks which is based on clustering techniques. The proposed Clustering Oriented - Earliest Available Time Scheduling** (CO-EATS) **creates groups of nodes whose messages are destined to common destination nodes. The goal of** CO-EATS **is to prevent consecutive messages from being destined to the same node. The simulation results have shown that the proposed scheme improves channel utilization and as a result it leads to higher network throughput while it keeps mean packet delay at low levels in comparison with conventional scheduling algorithms.**

I. INTRODUCTION

Message sequencing and channel assignment are two important issues in designing MAC protocols for optical Wavelength Division Multiplexing (WDM) networks [1]. A well-known, efficient scheduling algorithm for local area WDM networks with broadcast and select architecture is the Earliest Available Time Scheduling (EATS) [2]. EATS addresses the channel assignment without, however, handling message sequencing since it schedules messages according to their arrival order and ignores the fact that the messages' service order may affect the network's performance. This paper introduces a new algorithm that deals with the message sequencing issue based on the clustering [3], [4] of the network's nodes. The proposed Clustering Oriented - Earliest Available Time Scheduling (CO-EATS) organizes the network's nodes into clusters according to the destination of their messages. Then, given that each cluster will consist of nodes with probably the same destination, the CO-EATS defines the message sequencing choosing for transmission nodes from different clusters. In this way, it decreases the probability of scheduling messages to the same destination at successive order. As a result, the schedule length is reduced and the network performance is upgraded.

The proposed algorithm is inspired by the observation that consecutive messages to the same destination node may not fully use the available channels when the EATS algorithm is employed. Thus, it was necessary to enhance the EATS scheme with an efficient message sequencing mechanism which would distinguish consecutive messages destined to the same node. Clustering source nodes on the basis of their destination provide us with this mechanism since a clustering process, in general, aims at creating groups (i.e. clusters) of items that are similar between them and dissimilar to the items belonging to other groups. Therefore, discovering groups of nodes with common message destination and scheduling their messages properly could lead to higher network performance without aggravating the mean packet delay. Clustering has already been used in many domains, and especially on the Web, aiming at improving Web applications [5], [4].

The remainder of this paper is organized as follows. Section II provides the network background while clustering background is given in Section III. Section IV presents our new scheduling algorithm while Section V discusses the simulation results. Finally, conclusions and future work insights are given in Section VI.

II. NETWORK BACKGROUND

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). Given that each of the n network's nodes can either transmit a message or receive more than one messages, the sets $S = \{s_1, \cdots, s_n\}$ and $D = \{d_1, \cdots, d_n\}$ denote the source and destination nodes (e.g. s_i and d_i, where $i = 1, \cdots, n$, refer to the same node which in the first case behaves as source while in the second case as destination node). Each source node (s_i) is provided with a fixed-tuned transmitter (FT) for the control channel and with a tunable transmitter (TT) for the data channels. These are connected to a 2 x 1 combiner before reaching the n x n passive star coupler via an optical fibre. The n outputs of the star coupler are connected via n separate fibres to the destination nodes (d_i) which are equipped with 1 x 2 splitters that separate the data from the control channel. The control channel is connected to a receiver that is fixed-tuned to this channel (FR) while the data channel is led to a tunable receiver (TR) capable of tuning over the whole range of available data channels. Hence the system is CC-FTTT-FRTR as it is depicted in Fig. 1.

In the above CC-FTTT-FRTR implementation each transmission frame is divided into two phases namely the control and data phase. During the control phase a source node s_i

1-4244-1370-2/07/$25.00 ©2007 IEEE

Symbol	Definition
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 x 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
S	The w x t scheduling matrix
Cl	Clustering process
noc	Number of clusters
C_j	Cluster, $j = 1, \ldots, noc$
c_j	Cluster representative
Means	The noc x n clusters representatives' message table
d_E	Nodes distance over their messages' destination
J	Objective function

TABLE I

BASIC SYMBOLS NOTATION

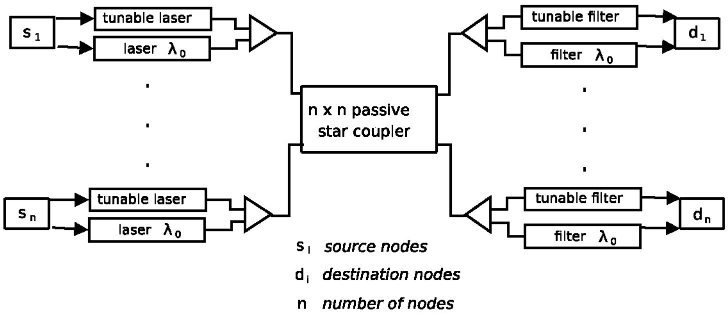


Fig. 1. Network Architecture

sends its control packet to the common control channel in a TDM-fashion while 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. In such a network, it is obvious that two or more source nodes might cause either channel collision, transmitting messages on the same data channel simultaneously, or receiver collision, transmitting messages destined to the same node simultaneously. Thus, in order to avoid collisions two tables are used on each node, namely the Receiver Available Time (*RAT*) and the Channel Available Time (*CAT*) tables. The *RAT* table consists of n elements, where $RAT(d_i) = t$, $i = 1, \cdots, n$, implies that destination node d_i will be available after t timeslots. The *CAT* table consists of w elements, where $CAT(i) = t$, $i = 1, \cdots, w$, denotes that channel i will be available after t timeslots. *RAT* and *CAT* are needed to avoid receiver and channel collisions respectively. A MAC protocol handles the above issues and runs a scheduling algorithm at the end of the control phase in each frame [6].

A well-known scheduling algorithm for such a network is the Earliest Available Time Scheduling (EATS) [2]. The core idea of EATS is to assign a message to the data channel that has the earliest available time among all the network data channels. Once the data channel is assigned, the algorithm proceeds to the message schedule as soon as that channel becomes available. EATS uses the *RAT* and *CAT* tables in order to keep a record of the channels and receivers state. With this global information in each node, the distributed EATS operates as follows: transmit a control packet on the control channel; select the channel with the earliest available time; define the transmission schedule based on *RAT* and *CAT*; and update these tables according to the last scheduled message. The algorithm produces the w x t scheduling matrix S, where t denotes the length on the schedule in timeslots. Each $s(i, j)$ element, $i = 1, \ldots, w$ and $j = 1, \ldots, t$, represents the destination node that receives a message on channel λ_i during the timeslot l_j. The time complexity of EATS is $O(nw)$.

III. CLUSTERING BACKGROUND

For the clustering process, the sets S and D are organized into an nxn message table M, whose $m(i, j)$ element, $i = 1, \ldots, n$ and $j = 1, \ldots, n$ indicate the length of the message from the source node s_i to the destination node d_j. Given that each s_i node can transmit a message per frame, it is obvious that the ith row of the M table will have one non-zero value. On the other hand, the jth 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. Under this notation, each node s_i is considered to be a multivariate vector consisting of n values and could be denoted 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) $C_1, \cdots, C_{noc}$, that is, $\bigcup_i^{noc} C_i = S$ and $C_i \bigcap C_j = \emptyset$ for all $i \neq j$. The noc clusters $C_1, \cdots, C_{noc}$ consist of $|C_1|, \cdots, |C_{noc}|$ 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.

Thus, the notion of similarity is fundamental in a clustering process, and so far it is quite common to evaluate the dissimilarity between two items (in our case the source nodes) by using a distance measure [3]. To proceed with the clustering of S, we employ the Squared Euclidean distance[1] which is a well-known and widely used distance measure in the vector-space model [3], [4]. Therefore, the evaluation of the dissimilarity between two source nodes e.g. $s_x, s_y \in S$ can be expressed by the distance of their vectors. Therefore, $d_E(s_x, s_y)$ denotes

[1]The Squared Euclidean distance uses the same equation as the Euclidean distance, but does not take the square root. For two points $X = (x_1, \cdots, x_n)$ and $Y = (y_1, \cdots, y_n)$ in n-space their Squared Euclidean distance is defined as: $\|x_i - y_i\|^2$

the Squared Euclidean distance of the nodes' vectors $M(x,:)$ and $M(y,:)$:

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

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 $dist(s_i, c_j)$. 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:

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

Based on the above we can define the network nodes clustering as follows: Given a network with a set S of n source nodes whose messages to n destination nodes (set D) are organized in an n x n message table M, the integers noc and k, and the objective function J, find a Cl clustering of S into noc clusters such that the J is minimized. A Cl that minimizes J groups together nodes from the set S that probably destine their messages to the same nodes of the set D.

IV. The Proposed Algorithm

The proposed CO-EATS is a two-step process which firstly handles the message sequencing and then deals with channel assignment based on the EATS algorithm. The core idea is that message sequencing should take into account the messages' destination. The proposed algorithm aims at grouping together nodes from S with the same destination. The goal is that messages to the same destination should not be scheduled in a successive order. Thus, CO-EATS schedules in sequence messages from nodes belonging to different clusters. Furthermore, CO-EATS prioritizes clusters as well as the members on each clusters according to the length of their messages.

More specifically, during the first step, we employ the K-means, a widely used partitional clustering algorithm [7], in order to produce the Cl clustering of S. Then, given the Cl and the message table M, we sort the members on each cluster according to the length of their messages. Similarly, using the *Means* table, consisting of the clusters representatives' vectors $\mathit{Means}(j,:)$, the *SortedC* is computed in order that we prioritize the clusters with longer messages. The calculated *SortedM* and *SortedC* are then used in order that the message sequencing will be defined. Once the *MsgSequencing* is formed, the algorithm proceeds to the second step called the channel assignment step. The goal of the function *ChannelAssignment* is to form the scheduling matrix S using the EATS algorithm.

Algorithm 1 The CO-EATS flow control

Input: A set S of n nodes organized in an nxn message table M, the upper bound on nodes' requests k and the number of clusters noc.

Ouput: The scheduling matrix S.

1: */*Clustering Step*/*
2: $(Cl, Means) = K - means(M, noc)$
3: $SortedM = \mathit{Quicksort}(M, Cl)$
4: $SortedC = \mathit{Quicksort}(Means)$
5: $MsgSequencing = \mathit{Sequencing}(SortedM, SortedC)$
6: */*Channel Assignment Step*/*
7: $S = \mathit{ChannelAssignment}(MsgSequencing)$

Theorem 1: The CO-EATS has time complexity $O(nlogn + nw)$.

Proof: During the clustering step we employ the K-means algorithm (line 2) whose time complexity is $O(n\ noc\ r)$, where n is the number of nodes, noc 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 to the number of nodes n and thus their contribution to the algorithm's complexity can be ignored [3]. Thus, the Cl clustering is computed in time linear on the number of nodes: $O(n)$. The *Quicksort* functions (lines 3 and 4) sorts the nodes and clusters' representatives in $O(nlogn + noc\ log(noc))$ time. The *Sequencing* function (line 5) takes time $O(n\ noc)$ to arrange the messages from the n nodes according to the $SortedM$ and $SortedC$. The total time complexity of the clustering step is thus $O(n + nlogn + noc\ log(noc) + n\ noc)$ which becomes $O(nlogn)$ since noc is relatively small compared to the number of nodes n. During the second step, the *ChannelAssignment* function (line 6) needs $O(nw)$ time [2] to form the scheduling matrix S, where w is the number of channels. As a result, the total complexity of CO-EATS is $O(nlogn + nw)$. ■

To facilitate the comprehension of the proposed scheme, let us consider a network consisting of the source nodes $(s_1, s_2, s_3, s_4, s_5, s_6)$, the data channels $(\lambda_1, \lambda_2, \lambda_3)$ and having the upper bound of nodes' messages length $k = 4$ packets. Then, a 6x6 message table M could be the following:

$$M = \begin{pmatrix} 0 & 0 & 4 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 2 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 2 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Example 1. In the above message table M the fact that $M(1,3) = 4$ means that the source node s_1 sends a message on length 4 to the destination node d_3. □

Applying the K-means for $noc = 3$ in the above M table results in $Cl = (3, 1, 1, 2, 2, 2)$ which can be represented by the following *Members* table:

C_1	s_2	s_3	
C_2	s_4	s_5	s_6
C_3	s_1		

TABLE II

THE TABLE *Members* BEFORE MEMBERS' SORTING

From Table II, it holds that $s_1 \in C_3$, $s_2, s_3 \in C_1$ while $s_4, s_5, s_6 \in C_2$. It is obvious that Cl places to the same cluster similar source nodes in terms of their destination nodes. Then, sorting the members on each cluster according to the length of their message results in swapping the nodes of C_1. Therefore, the above table is updated as follows:

C_1	s_3	s_2	
C_2	s_4	s_5	s_6
C_3	s_1		

TABLE III

THE TABLE *Members* AFTER THE MEMBERS' SORTING

Given this Cl the *Means* table is:

$$\textit{Means} = \begin{pmatrix} 0 & 0 & 0.5 & 1 & 0 & 0 \\ 0 & 1.67 & 0 & 0 & 0 & 0 \\ 0 & 0 & 4 & 0 & 0 & 0 \end{pmatrix}$$

Sorting *Means* provides our algorithm with the following service order: C_3, C_2, C_1. To this point, given that each cluster consists of nodes with probably the same destination, our scheme should separate them taking at the same time into account the result of *Means* sorting. Therefore, the message sequencing is defined as $s_1, s_4, s_3, s_5, s_2, s_6$ instead of the sequential one $s_1, s_2, s_3, s_4, s_5, s_6$. Tables IV and V depict the scheduling matrix S produced respectively. Based on these tables, the CO-EATS provides 22.2% improvement on channels' utilization while it reduces the mean packet delay from 3 to 2.2 timeslots.

	Timeslots						
	l_1	l_2	l_3	l_4	l_5	l_6	l_7
w_1	d_3	d_3	d_3	d_3			d_2
w_2	d_2	d_2		d_2	d_2		
w_3	d_4	d_4				d_3	

TABLE IV

THE SCHEDULING MATRIX S PRODUCED BY CO-EATS

V. EXPERIMENTATION

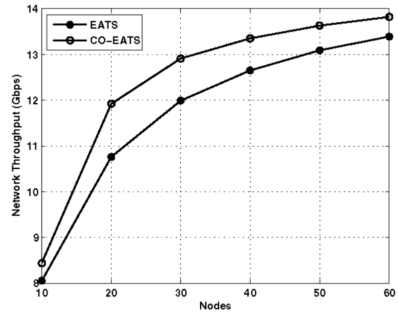
To evaluate the proposed CO-EATS in comparison with the EATS algorithm we carried out experiments which are based on the following assumptions:

1) The transmitters/receivers tuning time is set to 1 and the propagation delay of messages is set to 2.

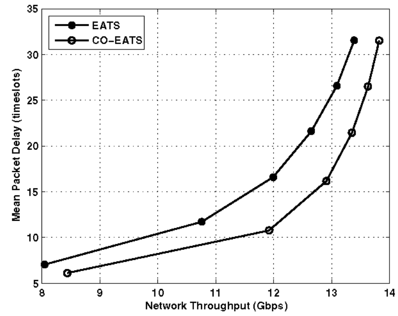
	Timeslots								
	l_1	l_2	l_3	l_4	l_5	l_6	l_7	l_8	l_9
w_1	d_3	d_3	d_3	d_3		d_2	d_2		
w_2						d_3			
w_3	d_4	d_4	d_2	d_2					d_2

TABLE V

THE SCHEDULING MATRIX S PRODUCED BY EATS



(a) Network throughput as a function of the number of nodes

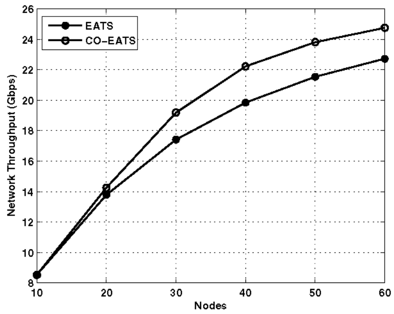


(b) Mean packet delay as a function of the network throughput

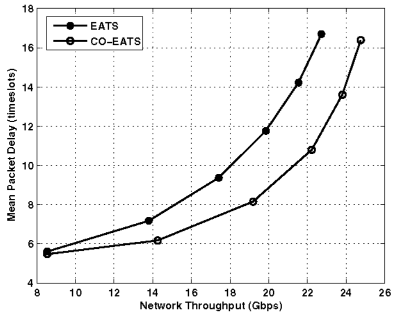
Fig. 2. Simulation results for $w = 5$ and $noc = 5$

2) The message transmitted by a node can be destined to every other node with equal probability.
3) Nodes may send messages of 0 to k length on each frame following uniform distribution.
4) The line is defined at 3 Gbps per channel.
5) The outcome results from 10000 transmission frames.

The performance of CO-EATS is compared to that of EATS in terms of the network throughput and mean packet delay. Network throughput represents the average number of bits transmitted per frame on each channel while mean packet delay denotes the mean time that messages are waiting to transmit. Fig. 2(a) and 3(a) depict the network's throughput as a function of the number of nodes for $n = 10, 20, 30, 40, 50, 60$ and $k = 10$ while the number of channels and clusters is $w = 5, noc = 5$ and $w = 10, noc = 10$ respectively. Defining $noc = w$ we succeed in not scheduling consecutive messages to the same destination since we choose to transmit messages from nodes belonging to different clusters which they probably have different destinations. It is apparent that



(a) Network throughput as a function of the number of nodes



(b) Mean packet delay as a function of the network throughput

Fig. 3. Simulation results for $w = 10$ and $noc = 10$

CO-EATS provides steadily higher throughput than EATS both for $w = 5$ and $w = 10$. Their maximum observed difference is 2.38 Gbps for $n = 40$ and $w = 10$ while for $n = 10$ and $w = 10$ the minimum difference is expected since each node destines its message to different channel and thus the contribution of clustering is of no value. In Fig. 2(b) and 3(b) we can observe that the improvement in network's throughput does not affect the mean packet delay. More specifically, the CO-EATS keeps lower the mean packet delay in comparison with EATS independently of the number of channels while obtains higher throughput. For example, for $n = 30$ and $w = 10$ CO-EATS offers 19.19 Gbps throughput causing 8 timeslots as mean packet delay while the respective values for EATS are 17.42 Gbps and 8.

VI. Conclusions and Future Work

This paper introduces and evaluates a novel message scheduling algorithm for WDM star networks which address both the message sequencing and channel assignment issues. The proposed Clustering Oriented - Earliest Available Time Scheduling (CO-EATS) deals with the message sequencing using a clustering approach which aims at grouping together network's nodes that sent their messages to common destination nodes. Based on the produced clusters the CO-EATS manages to avoid scheduling consecutive messages to the same destination which harms the channels' utilization. The proposed algorithm has been evaluated under uniform traffic for different number of nodes and channels and it has resulted in significantly upgrading the network performance while keeping low the mean packet delay in comparison with the EATS.

Future work aims at studying the simulation results using different values of propagation delay as well as evaluating the proposed scheme under poisson traffic. Furthermore, we will compare the proposed scheme with other message scheduling algorithms which also address the message sequencing issue.

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] 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.
[4] S. Petridou, V. Koutsonikola, A. Vakali, and G. Papadimitriou, "A divergence-oriented approach for web users clustering," in *Proc. of International Conference on Computational Science and its Apllications (ICCSA '06)*, Glaskow, Scotland, May 2006, pp. 1229–1238.
[5] 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.
[6] P.Sarigiannidis, G.I.Papadimitriou, and A.S.Pomportsis, "A high throughput scheduling technique, with idle timeslot elimination mechanism," *IEEE Journal of Lightwave Technology*, vol. 24, no. 12, pp. 4811–4827, 2006.
[7] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, 2001.