A Token Based Distributed Algorithm for Medium Access in an Optical Ring Network

A. K. Turuk, R. Kumar, and R. Badrinath

Department of Computer Science and Engineering
Indian Institute of Technology Kharagpur
Kharagpur-721302
{akturuk,rkumar,badri}@cse.iitkgp.ernet.in

Present Email akturuk@nitrkl.ac.in

Abstract. In this paper we propose a token based algorithm to access the shared medium in an optical ring network. The algorithm is based on reservation scheme. However, unlike other reservation schemes, which operate on three stages viz. reserve, transmit and release, the proposed scheme operates in two stages and does not explicitly release the resources. The proposed algorithm selects the earliest available data-channel for reservation hence we call it *Earliest Available Channel* (EAC) algorithm. The EAC algorithm operates in a distributed manner. Each node in the network maintains three modules: *send*, *receive* and *token processing* modules. The algorithm has the capability of handling channel collision and destination conflicts. We study the performance of the algorithm by simulation for bursty traffic modeled by M/Pareto distribution.

1 Introduction

It is widely acknowledged that the rapid growth in demand for bandwidth due to Internet explosion can be satisfied by optical networks, and in particular using the wavelength-division multiplexing (WDM) technology. A single fiber can support hundreds of wavelength channels. With the successful deployment of WDM in core networks, the access networks, viz., local area networks (LANs) and metropolitan area networks (MANs) are bottlenecks. Recently a lot of work has been reported in the literature for the deployment of WDM technology in the access network. Most of the work on LAN reported in the literature employ either a star or a ring as the underlying physical topology of the network. In a LAN the available bandwidth is shared among all the network users. To deal with multiuser access a media access control protocol is needed in such networks. In recent years many media access control protocols have been proposed for WDM-LAN based on star and ring as the underlying physical topology.

There are three well-known access strategy for LAN based on ring topology viz. token ring, slotted ring, and insertion register ring. These have been widely used as local area networks both in commercial system and research prototypes. Ring networks offer several attractive features such as higher channel utilization

S. R. Das, S. K. Das (Eds.): IWDC 2003, LNCS 2918, pp. 340–349, 2003.

and bounded delay. WDM slotted rings are reported in [1],[2],[3],[4]. Synchronization among the slots is a major design factor in slotted rings. Nodes must be synchronized so that a slot starts at the same time on all the wavelength channels in the network at the synchronization point. Synchronization points are the network hub for star topology and the WDM, ADMs for ring topology [5].

Token based WDM ring network is explained in [6],[7]. Unlike *FDDI* rings, authors in [6],[7] discussed multiple tokens in the ring. The number of tokens in the ring, the number of transmitter and receiver that each node is equipped with is equal to the number of data-channels available in the ring.

In this paper, we propose a token based algorithm which we call *Earliest Available Channel* (EAC) algorithm to access the shared medium in a WDM ring network. The algorithm is based on a reservation scheme.

Unlike other reservation schemes that operate in three stages viz. reserve, transmit and release, the EAC algorithm operates in two stages viz. reserve and transmit. In our proposed scheme, reserved resources are not explicitly released. Each node in the network maintains status of it's transmitter, receivers of other nodes and data-channels in the network. Status gives the time at which transmitter, receivers and data-channels are available. Resources (source node transmitter, receiver of destination node and a data-channel) are reserved for a duration which is determined at the time reservation request is made. The duration for which resources are reserved is different for different reservation requests. The reserved resources can be requested for reservation by another node after that period. This does not necessitate the explicit release of reserved resources. Two different nodes can make reservation requests for the same resource during the same cycle of the token but for different times. Transmitter of the source and receiver of the destination are tuned to the same reserved data-channel before communication between them takes place. In other words a lightpath is dynamically established between the source and destination along the reserved data-channel and remains in place until the transmission is completed. Availability of fast tunning lasers makes possible to set up lightpath dynamically.

The EAC algorithm operates in a distributed manner. Each node in the network maintains three modules: *send* module, *receive* module and *token processing* module. A node invokes 'send' module if its *req_made* queue (a queue stores all the reservation requests made by the node) is non-empty. Similarly, *receive* module is invoked if its *req_rec* queue (a queue stores all the transmission requests to the node) is non-empty. *Token processing* module is invoked when a node receives a token. *Req_made* and *req_rec* queues are updated by the *token processing* module. The reservation mechanism is explained in detail in the subsequent sections.

The algorithm has the capability of avoiding channel collision and destination conflicts. We study the performance of the algorithm by simulation for bursty traffic. We compare the performance with another token based algorithm *Multi-Token Inter-Arrival Time* (MTIT) Access Protocol [7]. To the best of

Table 1. Comparison of MTIT with EAC

	MTIT	EAC
Number of Tokens	Equal to the number of data-channel	One
Number of Transmitters and Receivers per node	Equal to the number of data-channel	Two
Fiber-Delay Lines	Exist at every node	No
Channel selection strategy	Selects a free channel	Selects the earliest available channel
Transmission	Simultaneous transmission on each data channel is possible at a node	A single transmission on a data-channel takes place at a node
Header processing	Header is processed at each intermediate node	No processing of header takes place
Packet Removal	Removed by the source	Removed at the destination
Token arrival	Inter-arrival of token at a node may differs	Inter-arrival of token at each node remains same

our knowledge, MTIT is the only token based protocol proposed for optical ring network. A qualitative comparison of MTIT and EAC is given in Table-1.

The rest of the paper is organized as follows. In Section 2 the system model is described. The EAC algorithm is described in Section 3. Simulation results comparing the proposed algorithm with MTIT are reported in Section 4. Finally, some conclusions are drawn in Section 5.

2 System Model

2.1 Assumptions

We consider a WDM ring network with N nodes. The system supports W wavelengths λ_0, λ_1, ..., λ_{W-1}. There are $W-1$ data channels and one control channel. One of the wavelengths, λ_0, is dedicated to control channel, and rest of the wavelengths are used as data channels. A circuit is established on wavelength λ_0 between every pair of adjacent nodes i and j. The circuit thus established is the dedicated control channel.

Each node is equipped with a *fixed* transmitter/receiver, and a *tunable* transmitter/receiver. The fixed transmitters and receivers are tuned to wavelength, λ_0, to transmit and receive control information between adjacent nodes while

tunable transmitters and receivers are tuned to data channels as and when required. For two nodes in the network to communicate, tunable transmitter of the source node and tunable receiver of the destination node must be tuned to the same wavelength (data channel). The system has a *single* token that circulates around the ring on the control channel. The token consists of N sub-fields which we call *slots*; *slot* i is assigned to node i. We define a $TokenPeriod$ (TP) as the period between two successive receives of the token by a node. We calculate TP as $TP = R + N \times p$ where p is the processing delay of token at each node and R is the ring latency. Since TP is same for all nodes in the network, each node gets a fair chance to access the shared medium. Thus, the delay involved is bounded.

A node on receiving the token processes each slot, l, $(0 \leq l < N)$, to update its knowledge about node l in the network. Prior to communication between a pair of nodes, the source must reserve the destination and a data channel. A node reserves the destination and a data channel by writing the control information at it's allotted slot in the token. A node has $N - 1$ buffers for each destination node. Reservation mechanism is explained latter.

2.2 Notations and Definitions

$DAT[i]$: Earliest time at which the node i will be available for receiving
$CAT[i]$: Earliest time at which the data channel i will be available for transmission
τ_d : Destination available time,
τ_c : Channel available time,
t_u : Tuning time of the transmitter/receiver,
t_p : Average propagation delay between source and destination, and
current_time : Time at which an action is taken at a node.

req_made queue : This is the queue that holds the reservation request made by a node. The i^{th} request made by a node is stored at the i^{th} element of the queue. An element of the queue has the following fields : tt – time at which transmitter of the node starts tunning to a data channel, di – identity of destination node to which transmission will take place, dc – wavelength to which the transmitter of the node will be tuned to, td – duration for which transmission will take place.
req_rec queue : This is a sorted queue that holds the reservation request from other nodes for which the current node (here current node is the node that is processing the token) is the destination. For example say, there is a reservation request from node 1, destined to node 5. When node 5 receives the token, reservation request from node 1 is entered in its *req_rec queue*. No other node will make an entry of this request in its *req_rec queue*. Elements of the *req_queue* are same as that of *req_made queue*.
Status : Indicates the status of the node's transmitter (*BUSY/FREE*), and
Finish : Indicates the status of the node's receiver (*BUSY/FREE*).

Control information in a $slot_j(s, d, c, t_c, D)$ are:
s : value of one indicates node j is requesting for reservation and value of zero indicates no request is made by node j,
d : identity of the destination node requested for reservation,
c : identity of the data channel requested for reservation,
t_c : time at which receiver of the destination, and transmitter of the source starts tunning to data channel c, and
D : duration of transmission.

3 Algorithm

Each node maintains global status of other nodes indicating the time at which nodes are available for receiving data (the i^{th} index of the vector DAT indicates the time at which node i will be available for receiving data). Similarly, nodes also maintain global status of data channels indicating the time at which data-channels are available for transmitting. Unlike the traditional reservation scheme where resources are reserved only when they are free, our proposed scheme looks ahead to find at what time the required resources will be available and reserves the resources from that point of time. Thus, in our scheme the explicit release of reserved resources is not required. Upon receiving the token, a node first updates its global status DAT and CAT vectors maintained at its node. If its buffers are non-empty, it selects the burst with maximum waiting time and an earliest available data-channel. Nodes then make the reservation request by writing the control information in its allotted slot. Then the token is sent to its adjacent node. When a node receives back the token (i.e., after a period of TP), its reservation request is completed, and all the nodes have recorded the next availability of the requested resources in their global status DAT and CAT vectors. Before transmission, transmitter of the source and receiver of the destination are tuned to the same data-channel. In other words, a circuit (lightpath) is established between the source and Destination, and remains established for the period of transmission.

3.1 EAC Algorithm

Send Module.

if (req_made queue is Non-empty and $Status = FREE$) *do*
1 Remove the front element of the req_made queue. Let it be $req\_made(l)$
2 $Status \leftarrow BUSY$
3 **if** ($current\_time \geq req\_made(l) \cdot tt$) *do*
- Start tunning the transmitter to data channel $req\_made(l) \cdot dc$

4 **if** ($current\_time \geq req\_made(l) \cdot tt + t_u$) *do*
- start transmitting data to node $req\_made(l) \cdot di$

5 **if** ($current\_time \geq req\_made(l) \cdot tt + t_u + req\_made(l) \cdot td$) *do*
- $Status \leftarrow FREE$

end.

Receive Module.

if (req_rec queue is Non-empty and $Finish = FREE$) *do*

1 Remove the front element of the req_rec queue. Let it be $req\_rec(l)$

2 $Finish \leftarrow BUSY$

3 **if** ($current\_time \geq req\_rec(l) \cdot tt$) *do*
- Start tunning the receiver to $req\_rec(l) \cdot dc$

4 **if** ($current\_time \geq req\_rec(l) \cdot tt + t_u$) *do*
- Start receiving data

5 **if** ($current\_time \geq req\_rec(l) \cdot tt + t_u + req\_rec(l) \cdot td$) *do*
- $Finish \leftarrow FREE$

end.

Token Processing Module. When a *node* i receives a token it invokes the token processing module. The following steps are performed to process the received token.

1 Examine $slot_i(s, d, c, t_c, D)$ *if* $(s = 1)$ *do*
- $Slot_i(s \leftarrow 0)$
- Add the request of node i in its req_send queue
- $DAT[d] \leftarrow CAT[c] \leftarrow t_c + t_u + t_p + D$

2 For all $slot_j(s, d, c, t_c, D), j \neq i$ *do*
- *if* ($s = 1$ and $t_c + t_u + t_p + D > DAT[d]$) *do*
  - $DAT[d] \leftarrow t_u + t_c + t_p + D$
- *if* ($s = 1$ and $t_c + t_u + t_p + D > CAT[c]$) *do*
  - $CAT[c] \leftarrow t_u + t_c + t_p + D$

3 *if* (node i buffers are non-empty) *do*
- find a burst with maximum waiting time. Let the destination identity of the burst be, say, x.
- let k be the earliest available channel. That is $k \leftarrow \{m : CAT[m]$ is minimum for $m \leftarrow 1, \cdots W - 1$ }
- Set $\tau_d \leftarrow DAT[x]$, $\tau_c \leftarrow CAT[k]$
- Find $\tau \leftarrow max(\tau_d, \tau_c)$. This gives the earliest time at which both the destination node x and the data channel k are available and can be reserved by a node.
- **if** ($\tau < current\_time + TP$) $\tau \leftarrow current\_time + TP$
- Calculate the duration of the transmission D
- Write the control information in $slot_i(s \leftarrow 1, d \leftarrow x, c \leftarrow k, t_c \leftarrow \tau, D)$

4 Send the token to successor node

end.

Table 2. Contents of DAT and CAT at the nodes at time t

DAT[0] = 0, DAT[1] = 5, DAT[2] = 7, DAT[3] = 10
CAT[λ_1] = 5, CAT[λ_2] = 7

Table 3. Traffic matrix at time t

Node	0	1	2	3
0		$b(10,4)$	$b(4,3)$	
1	$b(20,4)$		$b(25,3)$	
2		$b(10,3)$		
3	$b(10,5)$			

3.2 Simulation of Algorithm

We illustrate the reservation process with a simulation. We consider a four node ring network. The number of wavelengths is assumed to be three; contents of DAT and CAT vectors at time t are shown in Table-2. Table-3 shows the traffic matrix at time t (in the present example value of $t \geq 5$). The entry $b(x, y)$ corresponding to row m and column n of Table-3 indicates node m has a burst destined to node n. Duration of transmission of the burst is indicated by x, and y indicates the time at which the burst has arrived at node m. We assume the following quanta of values for the parameters: TP is assumed to be 20, t_u to be 2, t to be 6; propagation delay between a pair of adjacent node to be 5, and the processing delay of token at each node is assumed to be negligible.

Let node 0 has the token at time t. Node 0 selects the burst with maximum waiting time i.e, burst destined to node 2, and a earliest available data channel i.e, λ_1. Then it calculates $\tau_d = 7$ ($DAT[2]$), $\tau_c = 5$ ($CAT[\lambda_1]$), $\tau = 7$ (max (τ_d, τ_c). The value of τ is less than $t + TP$ ($\tau < t + TP$) so the value of τ is set to $t + TP$ i.e., 26. Node 0 writes control information in $slot_0(s = 1, d = 2, c = \lambda_1, t_c = \tau, D = 4)$ of the token and sends it to its successor node 1. Node 1 on receiving the token updates the contents of DAT and CAT vectors as shown in Table-4.

Node 1 selects the burst destined to node 2 and the earliest available data-channel λ_2. Assign $\tau_d = 32$, $\tau_c = 7$, $\tau = 32$. When node 1 receives the token, the value of t is updated to $t + 5$ (i.e., the updated value is t + the propagation delay between adjacent nodes which we have assumed to be 5 in our example). Node 1 writes control information in $slot_1(s = 1, d = 2, c = \lambda_2, t_c = \tau, D = 25)$ and sends it to node 2.

Updated values of DAT and CAT vectors at node 2 are shown in Table-5. Request from node 0 and node 1 are entered in the *req_rec* queue of node 2.

Node 2 selects the burst destined to node 1 and data channel λ_1. It sets the following values: $\tau_d = 5$, $\tau_c = 28$, $\tau = 28$. The value of τ is given by : $\tau < t + TP$ so the value of τ is set to $t + TP$, i.e., 36. Node 2 writes control information in

Table 4. Contents of DAT and CAT at node 1

DAT[0] = 0, DAT[1] = 5, DAT[2] = 32, DAT[3] = 10
CAT[λ_1] = 32, CAT[λ_2] = 7

Table 5. Contents of DAT and CAT at node 2

DAT[0] = 0, DAT[1] = 5, DAT[2] = 59, DAT[3] = 10
CAT[λ_1] = 28, CAT[λ_2] = 59

Table 6. Contents of DAT and CAT at node 3

DAT[0] = 0, DAT[1] = 48, DAT[2] = 59, DAT[3] = 10
CAT[λ_1] = 48, CAT[λ_2] = 59

$slot_2(s = 1, d = 1, c = \lambda_1, t_c = \tau, D = 10)$, and then sends it to node 3. Updated values of DAT and CAT vectors at node 3 are shown in Table-6.

Node 3 selects the burst destined to node 0 and data channel λ_1. It assigns the values : $\tau_d = 0$, $\tau_c = 48$, $\tau = 48$. Node 3 writes control information in $slot_3(s = 1, d = 0, c = \lambda_1, t_c = \tau, D = 10)$ and then sends the token to Node 0.

When node 0 receives the token, and finds its reservation request has been granted it puts it's request in *req_made* queue.

4 Simulation Results

We evaluated the performance of the EAC algorithm through simulation. We included the performance results, in this paper, in terms of throughput and mean packet delay. Throughput is defined as the transmission time divided by transmission time plus scheduling latency. This is a measure of how efficiently data channels are utilized in the network. For simulation, we considered a 10 node optical ring network. The number of wavelengths is 5. Values of other parameters are chosen as follows: capacity of wavelength channel is assumed to the 1 Gb/s, length of the ring is fixed at 100 kms, processing time of token at each node is assumed to be 1 μs, tunning time of transmitter and receiver are assumed to 5 μs. Calculated value of TP is 510 μs.

We consider two cases in our simulation. First, we consider the burst arrival to follow a Poisson distribution; the burst size was increased in (in multiples of 50) for every run of the simulation. Next, we considere the arrival of burst to follow a Poisson distribution and the burst size to follow M/Pareto distribution. In every run of simulation, the number of bursts generated is such that the number of packets generated will be one million.

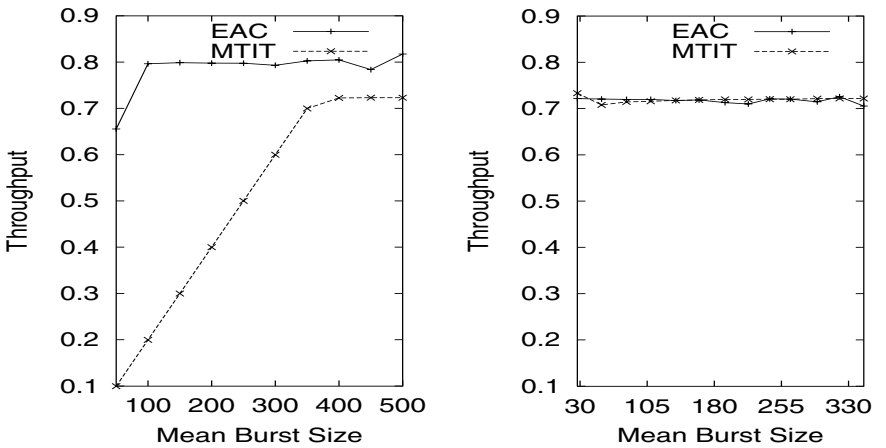

Fig. 1. Burst size *vs.* throughput for (a)fixed burst size, and (b) burst size taken from M/Pareto distribution for five data-channels.

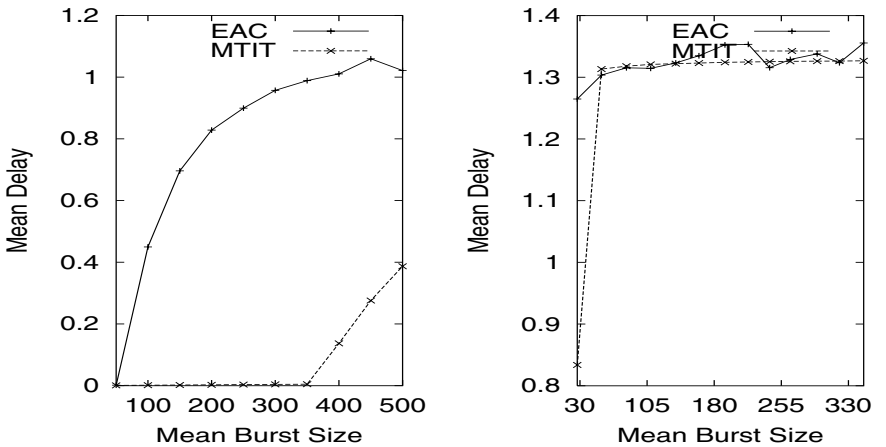

Fig. 2. Burst size *vs.* mean delay for (a) fixed burst size, and (b) burst size taken from M/Pareto distribution for five data-channels

4.1 Burst size *vs.* Throughput

Throughput increases with burst size for both MTIT and EAC as shown in Fig. 1. Wavelength utilization for fixed burst size is better in the case of EAC than MTIT as shown in Fig. 1(a). For burst size taken from M/Pareto distribution, wavelength utilization in EAC is almost identical to that of MTIT (Fig. 1(b)). This is the principal advantage of EAC algorithm that the wavelength utilization is identical with use of reduced resources.

4.2 Burst size *vs.* Mean Packet Delay

We define mean packet delay as the average end-to-end delay experienced by packets. For fixed burst size, delay experienced by packets are more in EAC than MTIT as shown in Fig. 2(a). However, for burst size taken from M/Pareto distribution, delay experienced by packets in both EAC and MTIT are almost identical for larger bursts (Fig. 2(b)).

5 Conclusions

In this paper, we proposed a token based distributed EAC algorithm for medium access in an optical ring network. EAC algorithm is based on a reservation scheme, and has the capability of avoiding channel collision and destination conflicts. We have compared EAC algorithm with another token based MTIT algorithm. MTIT algorithm is not scalable; the number of tokens required is equal to the number of data channels. We found that wavelength utilization is superior in EAC algorithm for bursts of fixed size. The utilization, however, is comparable in both EAC and MTIT algorithms for bursts taken from M/Pareto distribution. Delay experienced by packets are almost identical for both EAC and MTIT when the burst sizes are determined by M/Pareto distribution.

References

[1] Bengi, K., van As H. R.: Efficient QoS Support in a Slotted Multihop WDM Metro Ring. IEEE JSAC, **20** (2002) 216 – 227 341
[2] Marsan, M. A., Bianco, A., Leonard, E., Morabito, A., Neri, F.: All-Optical WDM Multi-Rings with Differentiated QoS. IEEE Commun. Mag., (1999) 58 – 66 341
[3] Marsan, M. A., Bianco, A., Leonardi, F. E., Toniolo, S.: An Almost Optimal MAC Protocol for All-Optical Multi-Rings with Tunable Transmitters and Fixed Receivers. 1997. http://citeseer.nj.nec.com/marsan97almost.html 341
[4] Marsan, M. A., Bianco, A., Abos, E. G.: All-Optical Slotted WDM Rings with Partially Tunable Transmitters and Receivers. http://citeseer.nj.nec.com/34986.html, 1996 341
[5] Spencer, M. J., Summerfield, M. A.: WRAP: A Medium Access Control Protocol for Wavelength- Routed Passive Optical Networks. Journal of Lightwave Technology **18** (2000) 1657 – 1676 341
[6] Fumagalli, A., Cai, J., Chlamtac, I.: A Token Based Protocol for Integrated Packet and Circuit Switching in WDM Rings. IEEE Globecom, Sidney (1998) 341
[7] Fumagalli, A., Cai J., Chlamtac, I.: The Multi-Token Inter-Arrival Time (MTIT) Access Protocol for Supporting IP over WDM Ring Network. Proc. ICC'99 conf. Vancouver, Canada, (1999) 341