DISTRIBUTED CONTROL IN OPTICAL WDM NETWORKS [*]

X. Yuan R. Gupta R. Melhem
Department of Computer Science
University of Pittsburgh
Pittsburgh, PA 15260

ABSTRACT

This paper describes and evaluates distributed wavelength reservation protocols for all–optical WDM networks. These protocols are essential for applying WDM techniques to large scale all–optical networks. The protocols ensure that the wavelengths on the links along a path are reserved before communication takes place. A message is transmitted using the reserved wavelengths and remains in the optical domain utill it reaches the destination. Based upon the timing at which the reservation is performed, the protocols are classified into two categories: forward reservation protocols and backward reservation protocols. Although forward reservation protocols are simpler, our performance study shows that backward reservation protocols provide better performance.

INTRODUCTION

Optical networks are rapidly gaining acceptance in computer and telecommunication applications due to their ability to offer large bandwidth. In optical networks, however, the signal transmission rates are much higher than the electronic processing speeds. The communication bottleneck is thus shifted from the transmission medium to the processing medium. *All–optical* communication schemes were proposed to reduce the impact of electronic processing on network control [3]. In an all–optical network, communications are carried out in pure circuit–switching fashion in the optical domain. No optical/electronic or electronic/optical conversion, and thus no electronic processing, is needed at intermediate nodes.

Wavelength–division multiplexing (WDM) in optical networks can provide multiple channels on each link by making different *virtual channels* communicate at different *wavelengths* [2, 3]. Two approaches, namely *link multiplexing* (LM) and *path multiplexing* (PM), can be used to establish connections in optical WDM networks. In WDM networks supporting LM, wavelength conversion devices are needed and a connection can be established if each link along the path has an available wavelength. Networks that support PM avoid the need for wavelength converters by establishing a connection only when there is a common available wavelength on all the links along the path.

In large scale optical networks, WDM techniques based upon centralized control are not feasible. Therefore, distributed wavelength reservation protocols are essential. Two types of protocols are considered and compared in this paper, namely *forward reservation* and *backward reservation* protocols. In the forward reservation protocol, the source node starts reserving wavelengths for the connection once it has a connection request. In the backward reservation protocol, once the source has a connection request, it sends a *probe* packet to the destination. The probe packet collects the wavelength usage information along the path to the destination. Upon receiving the probe packet, the destination node start reserving wavelengths along the path towards the source node. Although the protocols are described in this paper for PM, they can easily be modified to apply to LM.

Many studies of WDM in optical networks assume a wavelength assignment [1, 5]. Few works [6, 7] consider the on-line control mechanisms used to find that assignment. In [6], a distributed control algorithm to establish connections in multiplexed multistage networks is proposed. The work in [7] focuses on the comparison of the performance of PM and LM while taking into consideration the signaling overhead in the protocols. The protocols described in the above works fall into the *forward reservation* category. They are extensions of reservation schemes to establish connections in circuit–switching non–multiplexed networks [4]. The *backward reservation* schemes applied to multiplexed networks have not been described and evaluated before. Our study shows that, although backward reservation protocols are slightly more complex than forward reservation protocols, they consistently provide better performance.

In the next section, the wavelength reservation protocols are described, and in the following section performance results are presented. The last section concludes the paper.

CONTROL MECHANISMS

In order to support distributed wavelength reservation protocols, we assume that, in addition to the optical data network, there is a *shadow network* which is used to exchange control information. The shadow

---

[*] This work is supported in part by NSF award CCR-9157371 and by AFOSR award F49620-93-1-0023DEF.

network has the same physical topology as the data network. The traffic on the shadow network, however, consists of small control packets, and thus is much lighter than the traffic on the data network. The shadow network operates in packet switching mode; routers at intermediate nodes examine the control packets and update local bookkeeping information accordingly. The shadow network can be implemented as an electronic network, or alternatively, a virtual channel on the data network can be reserved exclusively for exchanging control messages.

Wavelength reservation protocols ensure that the path from a source node to a destination node is reserved before the connection is used. A path includes the wavelength on the links used by the connection, the transmitter at the source node and the receiver at the destination node. There are many options for wavelength reservation which are discussed next.

• *Forward reservation* versus *backward reservation.* Locking mechanisms are needed by the distributed protocols establishing a path to ensure the exclusive usage of a wavelength for a connection. This variation characterizes the timing at which the protocols perform the locking. Under forward reservation, the wavelengths are locked by a control message that travels from the source node to the destination node. Under backward reservation, a control message travels to the destination to probe the path. The wavelengths are then locked by another control message that travels from the destination node to the source node.

• *Dropping* versus *holding.* This variation characterizes the behavior of the protocol when it determines that a connection cannot be established. Under the dropping approach, once the protocol determines that the establishment of a connection is not progressing, it releases the wavelengths locked on the partially established path and informs the source node that the reservation fails. Under the holding approach, when the protocol determines that the establishment of a connection is not progressing, it keeps the wavelengths on the partial path locked for some period of time, hoping that during this period, the reservation will progress. If the reservation still does not progress, the partial path is then released and the source node is informed of the failure. Dropping can be viewed as holding with holding time equal to 0.

• *Aggressive* reservation or *conservative* reservation. This variation characterizes the protocol's treatment of each reservation. Under the aggressive reservation, the protocol tries to establish a connection by locking as much resources as possible during the reservation process. Under the conservative reservation, the protocol locks only a minimum amount of resources during the reservation process.

Network States

The control router at each node in the network maintains a state for each wavelength on each link emerging from that router. For a wavelength W on link L the state can be one of the following:

- $AVAIL$: indicates that the wavelength W is available and can be used to establish a new connection,
- $LOCK$: indicates that W is locked by some request in the process of establishing a connection.
- $BUSY$: indicates that W is being used in some connection to transmit data.

For the link, L, the set of wavelengths that are in the $AVAIL$ state is denoted by $Avail(L)$. When a wavelength, W, is not in $Avail(L)$, an additional field, CID, is maintained to identify the connection request locking W, if W is in the $LOCK$ state, or the connection using W, if W is in the $BUSY$ state.

Forward Reservation Schemes

Each connection request is assigned a unique identifier, *id*, which consists of the identifier of the source node and a serial number issued by that node. Each control message related to the establishment of a connection carries its *id*, which becomes the identifier of the connection, when successfully established. It is this *id* that is maintained in the CID field of locked or busy wavelengths on links. Four types of packets are used in the forward locking protocols to establish a connection.

• *Reservation packets* (RES) used to reserve wavelengths. In addition to the connection *id*, a RES packet contains a bit vector, *cset*, of size equal to the multiplexing degree. The bit vector *cset* is used to keep track of the set of wavelengths that can be used to establish a connection. These wavelengths are locked at intermediate nodes while the RES message progresses toward the destination node.

• *Acknowledgement packets* (ACK), used to inform source nodes of the success of connection requests. An ACK packet contains a *channel* field which indicates the wavelength selected for the connection. As an ACK message travels from the destination to the source, it changes the state of the wavelength selected for the connection to $BUSY$ and unlocks (change from $LOCK$ to $AVAIL$) all other wavelengths that were locked by the corresponding RES message.

• *Fail or Negative ack packets* ($FAIL/NACK$), used to inform source nodes of the failure of connection requests. While traveling to the source node, a $FAIL/NACK$ message unlocks all wavelengths that were locked by the corresponding RES message.

• *Release packets* (REL), used to release connections. A REL message traveling from a source to a destination changes the state of the wavelength reserved for that connection from $BUSY$ to $AVAIL$.

The protocols require that control messages from destinations to sources follow the same paths (in oposite directions) as messages from sources to destinations. We will denote the fields of a packet by $packet.field$. For example, $RES.id$ denotes the *id* field of the RES packet.

The forward reservation with dropping works as follows. When the source node wishes to establish a connection, it composes a RES message with $RES.cset$ set to the wavelengths that the node may use. This message is then routed to the destination. When an intermediate node receives the RES packet, it determines the next outgoing link, L, for the RES packet, and updates $RES.cset$ to $RES.cset \cap Avail(L)$. If the resulting $RES.cset$ is empty, the connection cannot be established, and a $FAIL/NACK$ message is sent back to the source node. This process of failed reservation is shown in Fig. 1(a). Note that if $Avail(L)$ is represented by a bit-vector, then $RES.cset \cap Avail(L)$ is a bit-wise *and* operation.

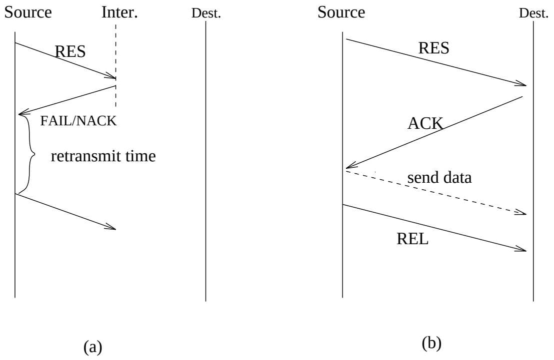


Figure 1: Control messages in forward reservation

If the resulting $RES.cset$ is not empty, the node reserves all the wavelengths in $RES.cset$ by changing their state to $LOCK$. It then forwards RES to the next node. This way, as RES approaches the destination, the path is reserved incrementally. Once RES reaches the destination with a non-empty $RES.cset$, the destination selects from $RES.cset$ a wavelength to be used for the connection and informs the source node of the wavelength to be used by sending an ACK message with $ACK.channel$ set to the selected wavelength. The source can start sending data once it receives the ACK packet. After all data is sent, the source node sends a REL packet to tear down the connection. This successful reservation process is shown in Fig. 1 (b).

Holding: The protocol described above can be modified to use the holding policy. Specifically, when an intermediate node determines that the connection for a reservation cannot be established, that is when $RES.cset \cap Avail(L) = \phi$, the node buffers the RES packet for a limited period of time. If within this period, some wavelengths in $RES.cset$ become available, the RES packet can then continue its journey. Otherwise, the $FAIL/NACK$ packet is sent back to the source. Implementing the holding policy requires each node to maintain a holding queue and to periodically check that queue to determine if any of the wavelengths have become available.

Aggressive and **conservative** reservation: The aggressiveness of the reservation is reflected in the size of the wavelength set, $RES.cset$, initially chosen by the source node. In aggressive reservation, the source node sets $RES.cset$ to $\{0, ..., N-1\}$, where N is the number of wavelengths in the system. On the other hand, the conservative reservation assigns $RES.cset$ to include only a single wavelength. Thus, aggressive reservation will be successful if there exists a wavelength that is available along all the links in the path while conservative reservation can be successful only when the specific wavelength in $RES.cset$ is available along all the links in the path. Although the aggressive reservation seems to increase the chance for a reservation to be successful, it results in overly locking the wavelengths. In the case of a heavy load, this might decrease the overall throughput.

Backward reservation schemes

In contrast to the forward reservation, backward reservation schemes use a forward message to probe the network, while locking the wavelengths is performed by a backward message. The backward reservation scheme uses five types of control packets, all of which carry the connection *id*, in addition to other fields as discussed next:

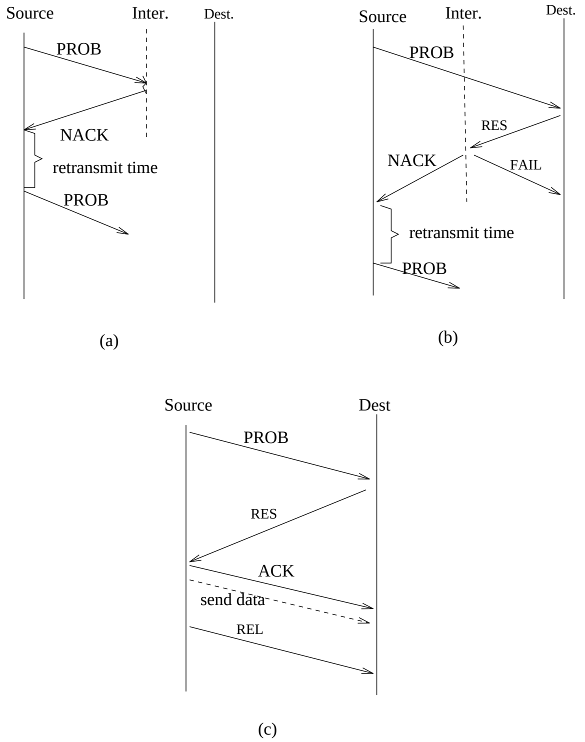
- *probe packets* (PROB), that travel from sources to destinations gathering information about wavelength usage without locking any wavelength. A PROB packet carries a bit vector, *init*, to represent the set of wavelengths available to establish the connection.
- *Reservation packets* (RES), similar to the RES packets in the forward scheme, except that they travel from destinations to sources, locking wavelengths as they go through intermediate nodes. A RES packet contains a *cset* field.
- *Acknowledgement packets* (ACK), similar to ACK packets in the forward scheme except that they travel from sources to destinations. It contains a *channel* field.
- *Fail packets* ($FAIL$), to unlock the wavelengths locked by the RES packets in cases of failures to establish connections.
- *Negative acknowledgement packets* ($NACK$), used to inform the source nodes of reservation failures.
- *Release packets* (REL), used to release connections after the communication is completed.

Note that a $FAIL/NACK$ message in the forward scheme performs the function of both a $FAIL$ message and a $NACK$ message in the backward scheme.

The backward reservation with dropping scheme works as follows. When the source node wishes to establish a connection, it composes a PROB message with $PROB.init$ set to contain all wavelengths in the system. This message is then routed to the destination. When an intermediate node receives the PROB packet, it determines the next outgoing link, L, for the PROB packet, and updates $PROB.init$ to

$PROB.init \cap Avail(L)$. If the resulting $PROB.init$ is empty, the connection cannot be established and a $NACK$ packet is sent back to the source node. Fig. 2(a) shows this failed reservation case.

If the resulting $PROB.init$ is not empty, the node forwards PROB to the next node. This way, as PROB approaches the destination, the wavelengths available on the path are recorded in the *init* set. Once PROB reaches the destination, the destination forms a RES message with $RES.cset$ equal to a selected subset of $PROB.init$ and sends this message back to the source node. When an intermediate node receives the RES packet, it determines the next link, L, for RES and updates $ACK.cset$ to $ACK.cset \cap Avail(L)$. If the resulting $RES.cset$ is empty, the connection can not be established. In this case the node sends a $NACK$ message to the source node to inform it of the failure, and sends a $FAIL$ message to the destination to free the wavelengths locked by RES. This process is shown in Fig. 2(b).



(a)

(b)

(c)

Figure 2: Control messages in backward reservation

If the resulting $RES.cset$ is not empty, the wavelengths in $RES.cset$ are locked and RES is forwarded to the next node. When RES reaches the source with a non-empty $RES.cset$, the source selects a wavelength from the $RES.cset$ for the connection and sends an ACK message to the destination with $ACK.channel$ set to the selected wavelength. This ACK message unlocks all the wavelengths locked by RES, except the one in *channel*. The source node can start sending data as soon as it sends the ACK message. After all data is sent, the source node sends a REL packet to tear down the connection. The process of successful reservation is shown in Fig. 2(c).

Holding: Holding can be incorporated in the backward reservation scheme as follows. In the protocol, there are two cases that cause the reservation to fail. The protocol may determine that the reservation fails when processing the PROB packet. In this case, no holding is necessary since no resources have been yet locked. When the protocol determines that the reservation fails during the processing of an RES packet, a holding mechanism similar to the one used in the forward reservation scheme may be applied.

Aggressive and conservative reservation: The aggressiveness of the backward reservation protocols is reflected in the initial size of *cset* chosen by the destination node. The aggressive approach sets $RES.cset$ equal to $PROB.init$, while the conservative approach sets $RES.cset$ to contain a single wavelength from $PROB.init$. Note that if a protocol supports only the conservative scheme, the ACK messages may be omitted, and thus only four types of messages (one packet per message) are needed.

PERFORMANCE EVALUATION

In the following discussion, we will use A to denote the aggressive reservation, C for conservative reservation, F for forward reservation, B for backward reservation, H for holding and D for dropping. For example, AFH denotes the aggressive forward reservation scheme with holding.

We have implemented a network simulator with various control mechanisms which include AFH, CFH, AFD, CFD, ABD, CBD, ABH and CBH. Some performance results for a 16×16 torus topology are given in this section. Fig. 3 shows the throughput that the protocols achieve assuming 10 wavelengths on each link. The time unit in the simulation is the time for a control packet to be processed at an intermediate node and propagate to the next node. This time is assumed to be also equal to the maximum time for a data packets to propagate from a source to a destination. Data messages are assumed to be 8 packets each, and a Poisson distribution for message arrival is assumed at each node. The holding time for the holding schemes are chosen to be 10 time unit, and the retransmit time is chosen randomly. As expected, the throughput of the network saturates for large message arrival rates. Fig 4 depicts the relation between the maximum throughput and the number of wavelengths under the same assumptions described above.

The results show that for 8-packet data messages, the conservative schemes outperform the aggressive schemes and the backward schemes outperform the forward schemes with the same aggressiveness. The conservative forward scheme, however, has higher throughput than the aggressive backward scheme. Holding improves the performance for all schemes except for the conservative forward scheme when the wavelength number is reasonably large.

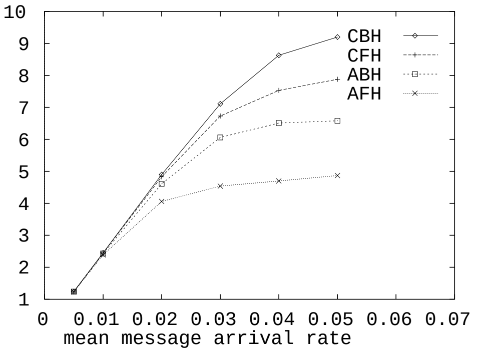


Figure 3: Throughput Vs. message arrival rate

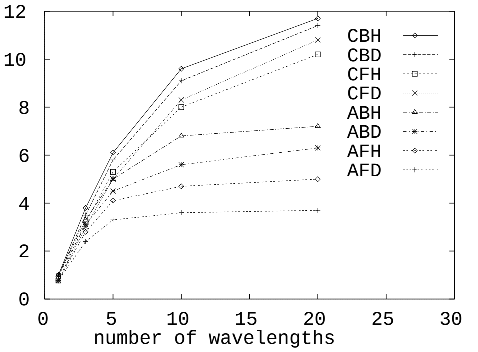


Figure 4: Maximum throughput

Tables 1 and 2 depict the effect of the message size on the maximum throughput. The throughput listed in the tables are normalized with respect to the throughput when each message is one packet long. Specifically,

$$normalized\ throughput = throughput \times msgsize.$$

The message size is found to have a large effect on the performance of the protocols. For small message sizes, the conservative schemes are significantly better than the corresponding aggressive schemes. However, for larger message sizes, the difference becomes smaller. Also for large message sizes, the backward scheme is always better than the forward scheme. The backward scheme always outperforms the forward schemes when the other options are the same.

msg size	normalized throughput			
	CFD	CFH	AFD	AFH
1	9.1	8.9	3.8	5.3
8	66.4	64.0	28.8	37.6
64	294.4	281.6	230.4	243.2
256	467.2	448.0	467.2	460.1
1024	579.1	563.0	576.7	572.2

Table 1: Forward reservation

msg size	normalized throughput			
	CBD	CBH	ABD	ABH
1	10.1	10.8	4.7	5.8
8	71.2	73.6	44.8	54.4
64	343.5	332.8	307.2	320.0
256	516.2	515.6	509.8	512.0
1024	597.9	603.9	598.0	612.5

Table 2: Backward reservation

CONCLUSION

Establishing a path in WDM multi-hop networks without wavelength conversion requires control protocols that are different from those used in circuit switching or ATM networks. In this paper, we have described various protocols to perform wavelength reservation in WDM networks. We examined forward and backward reservation protocols with variations of dropping and holding policies and aggressive and conservative policies. We are currently evaluating protocols that lie between the aggressive and the conservative extremes by optimizing the initial set sizes to achieve better performance.

References

[1] R.A. Barry and P.A. Humblet. "Models of Blocking Probability in All–optical Networks with and without Wavelength Changers." In *Proceeding of IEEE Infocom*, pages 402-412, April 1995.

[2] C. A. Brackett, "Dense wavelength division multiplexing networks: Principles and applications," *IEEE Journal on Selected Areas of Communications*, Vol. 8, pp. 948-964, Aug. 1990.

[3] I. Chlamtac, A. Ganz and G. Karmi. "Lightpath Communications: An Approach to High Bandwidth Optical WAN's." *IEEE Trans. on Communications*, Vol. 40, No. 7, July 1992.

[4] S. Nugent, "The iPSC/2 direct–connect communications technology." In *Proceedings of the 3rd conference on Hypercube Concurrent Computers and Application*, Volume 1, Jan. 1988.

[5] R. Ramaswami and K. Sivarajan, "Optimal Routing and Wavelength Assignment in All–Optical Networks." *IEEE Infocom*, vol.2, pages 970-979, 1994.

[6] C. Qiao and R. Melhem, "Reconfiguration with Time Division Multiplexed MIN's for Multiprocessor Communications." *IEEE Trans. on Parallel and Distributed Systems*, Vol. 5, No. 4, April 1994.

[7] C. Qiao and Y. Mei, "Wavelength Reservation Under Distributed Control." *IEEE/LEOS summer topical meeting: Broadband Optical Networks*, August 1996.