WDM Multicasting in IP over WDM Networks[*]

Chunming Qiao[†], Myoungki Jeong[†], Amit Guha[‡], Xijun Zhang[*] and John Wei[⋆]
[†]Departments of EE and CSE
University at Buffalo, Buffalo, NY 14260
[†]{qiao,mjeong}@cse.buffalo.edu
[‡]AT&T Labs, Lincroft, NJ 07738
[‡]amit@wnmail.wndev.att.com
[*]Lucent Technologies, Westford, MA 01886
[*]xijun.zhang@lucent.com
[⋆]Telcordia Technologies, Inc., Red Bank, NJ 07701
[⋆]wei@research.telcordia.com

Abstract

Supporting WDM multicasting in an IP over WDM network poses interesting problems because some WDM switches may be incapable of switching an incoming signal to more than one output interface. An approach to WDM multicasting based on wavelength-routing, which constructs a multicast forest for each multicast session so that multicast-incapable WDM switches do not need to multicast, was proposed and evaluated in [5]. Such an approach requires global knowledge of the WDM layer. In this paper, we study WDM multicasting in an IP over WDM network under the framework of Multiprotocol Label Switching (MPLS) [10, 11] using optical burst/label switching (OBS/OLS) [6]. We propose a protocol which modifies a multicast tree constructed by distance vector multicast routing protocol (DVMRP) into a multicast forest based on the local information only.

1. Introduction

As traffic demand in the Internet increases exponentially, Wavelength Division Multiplexing (WDM) networks ([1, 2, 3] with terabits per second bandwidth per fiber become a natural choice for the backbone in the future. A large body of research on WDM networks has been carried out, most of which studied unicast (i.e. one-to-one) communications. Recently, IP over WDM networks (or so-called Optical Internet) have received a considerable amount of attention [6, 7]. Given that multicasting (i.e. one-to-many communications) is important and increasingly popular on the Internet, issues concerning running multicast sessions in such IP over WDM networks need to be studied.

[*]This research is supported in part by NSF under contract ANIR-9801778 and by DARPA under contract F30602-98-C-0202.

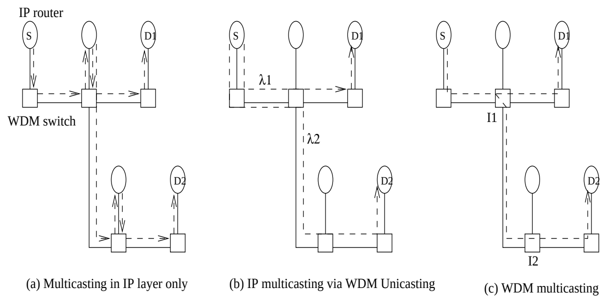
IP router
S
D1
D2
WDM switch
λ1
λ2
I1
I2
(a) Multicasting in IP layer only
(b) IP multicasting via WDM Unicasting
(c) WDM multicasting

Figure 1. Multicasting on IP over WDM networks.

Multicast sessions may run in IP over WDM networks by letting each IP router on a multicast tree make copies of a data packet and transmit a copy to each (dependent) downstream router (see Figure 1. (a)). However, this requires O/E/O conversion of every data packet at *all* the routers on the tree which may be inefficient (in terms of e.g. latency) and undesirable (in term of e.g. data transparency). Such O/E/O conversions may be avoided by using a virtual topology consisting of lightpaths (i.e. wavelength routes) from the multicast source to each destination of the multicast session (see Figure 1. (b)). However, for large multicast groups, the network bandwidth consumed by such a scheme may become unacceptable because of the unicasting nature of the lightpaths [5]. Accordingly, "true" multicasting at the WDM layer by taking advantage of the power-splitting (or multicasting) capability of the WDM switches is desirable (see Figure 1. (c)).

WDM multicasting has several potential advantages. First, with the knowledge of the physical (i.e. WDM layer) topology, which may differ from what is seen at the upper electronic (e.g. IP) layer, more efficient (in terms of bandwidth and/or latency) multicast trees can be constructed at the WDM layer. Secondly, some WDM switches uses power-splitting components, and power-splitting is more efficient than copying (by IP) for multicasting purposes. Finally, multicasting at the WDM layer provides a higher degree of data transparency (in terms of bit-rate and coding format).

Unfortunately, due to the lack of optical storage (and logic), some WDM switches is *multicast-incapable* (MI) by design (as to be discussed in Section 2). Accordingly, keeping the multicast data in the optical domain may not be easy. For example, if WDM switch I_1 in Figure 1. (c) is MI, that is, it cannot split the power of an incoming signal, the multicast tree constructed by IP must be modified to ensure that the multicast data can reach both D1 and D2. In this example, a possible solution is to use two separate lightpaths, making WDM multicasting as efficient (or inefficient) as IP multicasting based on lightpaths, which is shown in Figure 1. (b). However, even in the presence of MI switches, WDM multicasting will be, in general, much more efficient than IP multicasting based on lightpaths (or WDM unicasting) [5].

A network (IP over WDM) where only a fraction of WDM switches has the multicast capability is said to be *sparse splitting*. One way to support WDM multicasting in sparse splitting networks is to use a new or modified IP multicast protocol that constructs an appropriate IP multicast tree, where an IP router can serve as a branching node only if the router has an underlying WDM switch which is multicast-capable (MC). Such a tree can then be used by the WDM layer for switching the multicast data. An alternative is to keep the semantics of existing IP multicasting protocols intact but let the WDM layer modify the IP multicast tree constructed by DVMRP [4] (or MOSPF [12]) appropriately. This approach requires global knowledge of the physical (WDM) network including the multicast capability (or incapability) of each WDM switch. Such knowledge may be obtained by, e.g. letting the WDM layer "piggyback" the necessary information about the WDM layer onto the link-state-advertisements (LSAs) used by OSPF [13].

When the global knowledge of the WDM layer is not readily available, we must rely on a distributed protocol running at the WDM layer that uses only local knowledge (i.e. the information on the multicast-capability of a WDM switch itself). The rest of this paper is organized as follows. In Section 2, we examine WDM switch architectures. In Section 3, we discuss issues and general approaches related to WDM multicasting in IP over WDM networks. In Section 4, we present the proposed WDM multicasting protocol which uses optical burst/label switching (OBS/OLS) to interwork with DVMRP. Section 5 concludes the paper.

2. WDM switch architectures

In this section, we describe two types of WDM switch architectures: one is multicast-incapable (MI) and the other is multicast-capable (MC), and give an example for each. In general, a space-division (SD) switch in a WDM network, such as those made of electro-optic (e.g. $LiNbO_3$) directional couplers, is MI. A MC switch needs to use splitters (or passive couplers) as a component.

An example MI switch architecture is shown in Figure 2. In this architecture, each input WDM signal (or fiber) is demultiplexed first, and each channel may then be converted to a different wavelength using a wavelength converter (WC) in order to avoid contention. Each channel is routed to a desired output port by an NM by NM SD switch (which may be implemented in multiple stages using smaller SD switches). Using a larger SD switch having more than *NM* inputs and outputs, a channel can be dropped and added by the local router/host (not shown). Accordingly, in the rest of paper, it is assumed that a MI switch is still capable of "drop-and-continue" [5], that is, a signal carried on an incoming channel can be received by the local router and forwarded to *one* output channel. In addition, the switch connected to the multicast source (via the source router), even though it might be MI, will be able to transmit multiple "copies" of a signal to different outputs, or the same output port using the same wavelength (serially) or different wavelengths (concurrently).

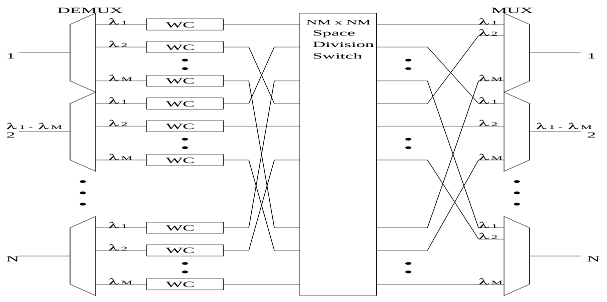

Figure 2. An example of multicast-incapable (MI) switch.

An example MC switch architecture is shown in Figure 3. Here, an input WDM signal is split into *NM* signals through two stages of splitters (first $1:N$ and then $1:M$), one WDM signal for each N by 1 SD switch at the third stage. Each SD switch then selects one of the *N* input WDM signals, out of which one wavelength is extracted using a tunable filter (TF). The extracted wavelength may then be converted into a different wavelength to avoid contention. To support multicasting, the same input WDM signal is selected by multiple SD switches, which are normally connected to different outputs. For instance, the two solid lines in Figure 3. shows how λ_1 of input port 1 multicasts to output 1 (using λ_2) and output 2 (using λ_M). Note that, this architecture is so flexible that it can even send multiple "copies" to the same output using different wavelengths by letting multiple SD switches be connected to the same output (e.g. SD switches 1, 2, and *M*) select the same input WDM signal.

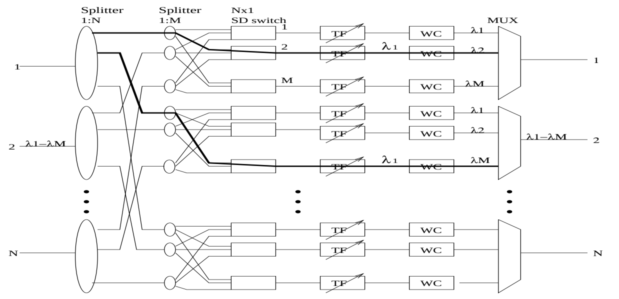

Figure 3. An example of multicast-capable switch.

In a sparse splitting network to be considered in the rest of the paper, it is assumed that a MC switch is similar to the one shown in Figure 3., which is strictly-nonblocking (even for multicasting).

3. Multicast in IP and WDM

DVMRP [4] is currently the dominant multicasting protocol used by the Internet (or specifically, the MBone). It is a data-driven multicast routing protocol, which constructs a source-based multicast tree (for each source-group pair) as follows. Multicast capable routers exchange DVMRP routing tables (see Figure 4. for an example) at intervals. When a source has multicast data to send, it is first broadcast to the entire network using Reverse Path Forwarding (RPF). Each router maintains a forwarding cache (see Figure 5.) which has, for each multicast session, the upstream router (from which it receives data) and all downstream routers (to which it must forward data). The multicast tree established by RPF is then *pruned* by removing routers that do not need to receive multicast data.

Source Network	Cost
128.2.3	10
132.45.3	20

Figure 4. DVMRP Routing Table.

After this broadcast-prune stage, multicast data is sent only to the routers on the multicast tree for a period of time. In order to receive multicast data during such a period, a router can send a *graft* message towards the source, and the graft message cascades up till it reaches a router on the multicast tree corresponding to that source (and group). Each router relaying the message makes an entry for the source-group pair in its forwarding cache.

Source Prefix	Group	Input Interface	output Interface
128.2.3	224.4.5.9	3	1 2 4
128.2.3	224.4.5.10	3	2

Figure 5. DVMRP Forwarding Cache.

Note that a IP multicast tree consists of only routers capable of multicast (called MC routers). Specifically, in IP, a router is incapable of multicast (called a MI router) only because the router does not understand (or is not running) the appropriate IP multicast protocol (note that a router can always make copies). In addition, IP-in-IP tunneling is used for exchanging data between MC routers, or in other words, a MI router cannot "drop-and-continue". On the contrary, when discussing WDM multicast in this paper, we assume that every WDM switch, MC or MI, understands (runs) whichever WDM multicasting protocol that we care to use. In addition, since a MC router may have a MI switch underneath it, a WDM multicast tree can (and in some cases, must) have MI switches. In fact, a WDM *multicast forest* [5] consisting of multiple trees may be necessary to include all the switches belonging to the same multicast group. For example, solid lines in Figure 6. show an multicast tree copied from the IP layer. In order to reach all the marked switches using WDM multicasting, a tree consisting of a single branch (the dashed line) from S to Y is needed *in addition to* a tree (shown in dotted lines) containing nodes A, B, X and Z. The above discussion clearly suggests that the problem in WDM multicasting introduced by having MI switches is different from that in IP multicasting introduced by having MI routers.

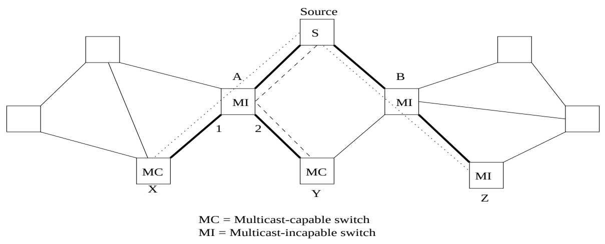
Source
S
A
MI
1
2
B
MI
MC
X
MC
Y
MI
Z
MC = Multicast-capable switch
MI = Multicast-incapable switch

Figure 6. A WDM multicast forest.

There are two WDM multicasting approaches, one based on wavelength-routing as in [5, 8, 9], and the other based on optical burst/label switching (OBS/OLS). In the former, multicast data will be switched to one or more outgoing wavelengths according to the incoming wavelength that carries it (as in wavelength-routing). In other words, a wavelength needs to be reserved on each branch of a multicast forest as in circuit-switching. For example, λ_1 may be reserved from S to X (and from S to Z), while λ_2 may be reserved from S to Y. This scheme is suitable for high bandwidth multicast applications having a relative long duration, such as video distribution.

In this paper, we focus on the alternative, which sets up virtual circuits (in the context of MPLS). Specifically, when using Optical Burst/Label Switching (OBS/OLS) to support WDM multicasting, each WDM switch maintains its own forwarding cache which is similar to the forwarding cache maintained by the IP router above it. In fact, the first four columns of a WDM forwarding cache at a switch are initially copied from the DVMRP forwarding cache (an example is shown in Figure 5.), except that if the switch is MI, the fourth column needs to be updated so there is only one output interface per row (see Figure 7.). In addition, two more columns for incoming and outgoing labels, respectively, are created. For the multicast forest shown in Figure 6., two rows will be created at the MI switch A with two different incoming labels (L_1 and L_2) for X (output 1) and Y (output 2), respectively.

Once the WDM forwarding cache is created, each burst (containing one or more IP datagrams) carries a short, fixed-size label using e.g. the subcarrier multiplexing technique. If the label matches an incoming label in the forwarding cache, the burst will be switched to the corresponding output interface with the (possibly different) outgoing label. Label processing (i.e. matching and modification) can be done either optically or, with O/E/O conversions, electronically. In any case, the payload remains in the optical domain by going through fiber-delay lines (FDLs) while its label is being processed. To avoid the use of FDLs, a control packet containing a label can be sent first to set up the switches, while a burst waits at the source for an offset time [6]. Either way, bursts can statistically share the bandwidth of all the wavelengths, making this approach more suitable for supporting multicasting applications with bursty and/or relatively low bandwidth traffic.

Source	Group	Input	Output	Incoming label	Outgoing label
S	g_1	3	1	L_1	L_3
S	g_1	3	2	L_2	L_4

Figure 7. WDM Forwarding Cache.

4. WDM multicasting protocol based on OBS/OLS

In this section, we describe a distributed protocol which, using only local knowledge of the WDM layer, modifies a multicast tree built by the DVMRP, so that MI switches need not multicast. The basic idea is as follows. Once the initial broadcast-prune stage is over (i.e. a multicast tree is constructed by the DVMRP), the source WDM switch initiates the modification of the tree (into a multicast forest) by sending a *repair* message to its downstream switches. A non-leaf MC switch simply relays the repair message to all its downstream switches (see Figure 8. (a)). On the other hand, a non-leaf MI switch relays the repair message to only one downstream switch and sends a *purge* message to all other downstream switches (see Figure 8. (b)). Any switch which has received the repair message is considered to be on the new multicast forest, and as this repair message propagates down the original multicast tree, the multicast forest will be constructed using label-switched paths (LSPs).

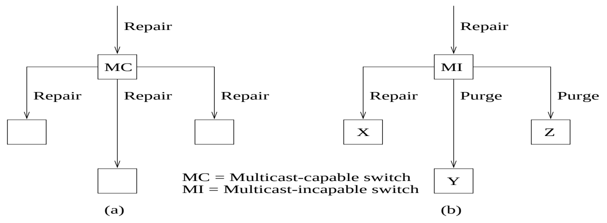
Repair
MC
Repair
Repair
Repair
MC = Multicast-capable switch
MI = Multicast-incapable switch
(a)
Repair
MI
Repair
Purge
Purge
X
Z
Y
(b)

Figure 8. Repair Messages.

A purged switch can always *grow* back (i.e. be attached) to the new multicast forest by establishing a new LSP all the way from the source through its parent (e.g. see the dashed line in Figure 6.). We will describe two grow schemes where a purged switch sends a *grow request* message to all its neighbors (including the parent switch which purged it) in order to find a more efficient LSP. In the *direct* grow scheme, a switch which was not on the original multicast tree will ignore any grow request, but in the *indirect* grow scheme, it can relay a grow request, and may become a node in the new multicast forest. In either scheme, a purged switch may grow back using a shorter LSP through e.g. a sibling switch, as to be described later.

Note that prior to the completion of the multicast forest construction, data is routed by IP (in much the same way as in Figure 1. (a)). Afterwards, data is switched at the WDM layer as in Fig. (c) based on the WDM multicast forest. In addition, to accommodate frequent graft messages without having to reconstruct the entire WDM multicast forest (i.e. without requiring the multicast source to send a repair message), the WDM switch (say S_N) under the IP router (say R_N) may intercept the graft message and send a grow request to establish a LSP from the existing multicast forest to switch S_N. Since such a LSP may use an input interface which is different from that expected by router R_N according to its RPF rule, switch S_N needs to alter the input interface associated with the first IP datagram so that an appropriate entry in the forwarding cache at router R_N and subsequently switch S_N can be created. A similar technique can be used to accommodate frequent prune messages but details are omitted. A more detailed description of the protocol follows.

4.1. Repair and purge

As mentioned earlier, if a MI switch receives a repair message and has more than one downstream switch (see Figure 8. (b)), it selects one downstream switch to relay the repair message to (e.g. switch X) and sends *purge* messages to all other downstream switches (e.g. Y and Z).

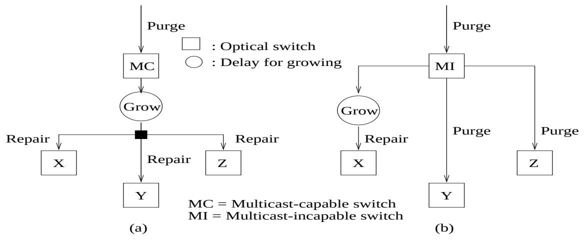

Figure 9. Purge Messages.

A purged switch with an attached *member* router (or host), i.e. one that belongs to the multicast group, must grow back, and can do so after it receives a reply (or replies) to its grow request (*grow reply* messages are to be discussed in the next subsection). Note that a purged switch (even if it is MC) can send repair messages only after it grows back (see Figure 9. (a) and (b)). This is because otherwise, its downstream switches will be able to reply to other grow requests before their parent (i.e. the purged switch) grows back, and hence may result in a loop (i.e. a group of switches that are disconnected from the multicast forest). Nevertheless, if the purged switch is MI, it may send purge messages to all but one downstream switch *before* it grows back in order to speed up the multicast forest construction process (see Figure 9. (b)).

A MI switch can either randomly or use some heuristics to select the downstream switch to send a repair message. For example, it may select a MC switch (instead of a MI switch) so that potentially the repair message can reach more descendent switches quickly, which in turn can speed up the construction process of the multicast forest. The MI switch may also purge all its downstream switches which are requested to report the cost for them to grow back through nodes other than their parent (i.e. the MI switch), and set a timer. If all downstream switches report back before the timer expires, the MI switch then selects one reporting the highest cost. Otherwise, it selects one whose report is not in yet (the node may not be able to grow back with a finite cost). The cost for a purged switch (e.g. P in Figure 10.) to grow back could be simply the number of hops (or the bandwidth consumption or latency) of a new LSP to be established. For example, if P's neighbor X is already on the multicast forest (see Figure 10. (a)), and is either a MC or a leaf MI switch, the cost for P to grow through X is one hop.

Note that although a purged switch without any attached member router/host does not have to grow back, to simplify the protocol and reduce signaling (or control) overhead, we propose that it will grow back as long as it has at least one downstream switch. This is because otherwise, the purged switch has to send one purge message to each of its downstream switches, which may then send their own purge messages, resulting a potentially large number of purge messages (and other subsequent control messages such as grow request messages to be described next).

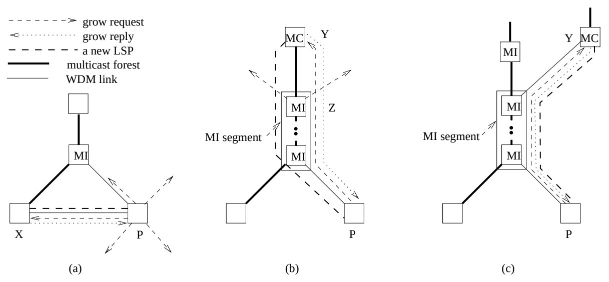

Figure 10. Grow Messages.

4.2. Grow

As mentioned earlier, there are two grow schemes, direct and indirect, and the only difference between the two is that in the former, a neighbor which was not on the multicast tree will simply ignore any grow request, while in the latter, it may relay the grow request. Note that even in the direct scheme, a new LSP from the source to a purged switch can always be established in the worst case. However, as shown in Figure 10., the purged switch P may grow back using a shorter (or lower-cost) LSP from its neighbor (e.g. switch X in Figure 10. (a)) or some ascendent MC switches (e.g. switch Y in Figure 10. (b) and (c)). In addition, since a MI switch (e.g. switch Z in Figure 10. (b)) receiving a grow request cannot reply directly unless it is a leaf switch, a grow request may need to be relayed even in the direct scheme. The indirect scheme is more complex than the direct scheme but may result in a more efficient multicast forest.

In either scheme, a purged switch can grow back as quickly and efficiently as possible by sending a *grow* request to each of its neighbors (see switch P in Figure 10. (a)). A switch relaying the request will also relay it to all its neighbors except the one from which the request came (see switch Z in Figure 10. (b)). To keep the number of *grow* requests in check, each request carries a time-to-live (TTL) value, which is initially set to no greater than the number of hops, say *h*, from its sender to the multicast source along the original multicast tree. The TTL value is decreased every time a request is relayed, and a request with a TTL value of 0 will not be relayed.

In addition, a request is relayed only if the receiving switch does not have any outstanding requests (i.e. has an empty request queue), and is incapable of replying (either because it is a non-leaf MI switch on the multicast forest, or because it was not even on the original multicast tree). If the receiving switch is incapable of replying but has an outstanding request (i.e. a non-empty request queue), the request will simply be queued for a time-out period (waiting for a reply). In case that the receiving switch is capable of replying (because it is a MC or a leaf MI switch on the multicast forest) – such as switch T in Figure 11., it sends a grow reply back with the cost to grow from T to Y (instead of from T to P).

As soon as a switch which has relayed a grow request (e.g. X or Y) receives a reply, it replies to *all* queued requests *in parallel* with an appropriately adjusted cost-to-grow. However, as illustrated in Figure 11., a purged switch P (which sent its own request) will wait either till all neighbors (to which it sent a request earlier) reply, or till there is a time-out with at least one reply. If there is no reply after a time-out, P has to resend a grow request with possibly a larger time-out value. If there is at least one reply, P selects the best neighbor from among the ones that have replied, and then sends a *grow confirm* message, which is essentially a "request" for a new LSP (in this case, from T to P) and thus may specify the outgoing label to be used by its upstream switch (in this case, X). After P sends the grow confirm message and creates an entry in its WDM forwarding cache (see Figure 7. and related discussion), P is considered to be on the multicast forest. Similarly, a switch, e.g. X or Y, which has not already been on the multicast forest, is considered to be on the multicast forest *only after* it has processed (and forwarded) a confirm message.

Note that if T in Figure 11. is a non-leaf MC switch, it can originate a new LSP by e.g. adding a new outgoing label specified by Y for an existing entry. If T is a leaf MI switch and this is the first confirm message it receives, the case is similar, except that it has to create a new entry afterwards and T becomes a non-leaf (MI) switch. Since a non-leaf MI switch cannot originate a new LSP, it has to forward any subsequent confirm messages it receives to its upstream switch.

When a MI switch (say X in Figure 11.) performs *parallel reply* as described above, it may not be able to reply to the second or later request in the queue with an accurate cost-to-grow in some cases. For example, assume that all the switches (e.g. X and Y) are MI, and T is MC, then the first as well as the second (or later) LSPs will originate from T and thus have the same cost for all requests queued at X. However, if T is a leaf MI switch, since the second (or later) LSP will originate from some switch beyond T, its actual cost to X will be larger. On the other hand, if X is

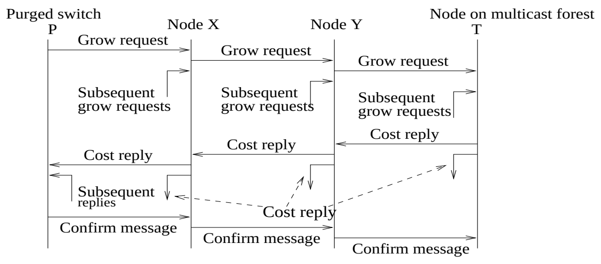

Figure 11. Grow request, reply and confirm messages.

MI, but some switch, say Y, is MC, (regardless of whether T is a MC, a leaf MI, or a non-leaf MI switch), the second or later LSP will originate from Y and hence has a lower actual cost. While a more elaborate protocol may provide an accurate cost-to-grow to each request, such a protocol will require *sequential reply*, whereby a MI switch replies to one request, and waits for its confirmation (or the absence of it) before it reply to the next. However, the increased complexity of the protocol and the delay in replying to grow requests are perhaps more undesirable than having inaccurate cost occasionally. In fact, the protocol may be simplified by letting a purged switch pick a neighbor from which a reply came in first (i.e. without using or comparing costs of the reply messages). In addition, a switch which is not already on the multicast forest may simply reject all grow requests (except that in the indirect grow scheme, such a switch will relay the first request).

4.3. State diagram

In this section, we summarize the main protocol features after making the above two simplifications, namely, earliest-reply first and no grow request queuing by using a state diagram and its corresponding transition table. It is assumed that after a broadcast-prune stage, every switch will go back to the INITIAL state, and the next broadcast-prune stage will start after the construction of the multicast forest is completed.

Figure 12. shows the states where a switch might be in during the process of constructing a multicast forest, as well as the transitions between the states using either the direct or indirect grow scheme. More specifically, in the direct grow scheme, only the solid lines represent possible transitions, and neither of the two states (namely, GROWING and GROW-REPLY) is valid. In the indirect grow scheme, all states and transitions (including those shown in solid and dashed lines), except transition 15, are valid. Each of the transitions has a triggering event and associated actions which are described in Figure 13.

Consider the state diagram in Figure 12. corresponding to the direct grow scheme first. After the broadcast-prune

Conditions	Events	Actions
1	member after broadcast-prune	copy DVMRP forwarding cache
2	not member after broadcast-prune	do nothing
3	receive grow request	ignore
4	receive purge message	flood grow request to all neighbors with a retry-timer
5	receive repair message	relay and if MI, send purge message to downstreams except one
6	time-out (retry-timer) without any reply	reflood grow request to all neighbors with a larger retry-timer
7	receive first grow reply from a node X within time-out	send confirm message to X and repair message to downstreams
8	receive grow request	if MC or leaf MI, send reply and set a confirm-timer; if non-leaf MI, flood to all but one
9	receive grow reply	if reply to itself, ignore; if non-leaf MI and first reply to another node Y, relay
10a	receive confirm message	update WDM forwarding cache (with a new LSP) and relay confirm message (only if it is a non-leaf MI)
10b	receive confirm message	update WDM forwarding cache (with a new LSP) and relay confirm message
11	time-out (reply-timer or confirm-timer)	discard related history/transition information
12	receive first grow request	flood it to all neighbors but one and set a reply-timer
13	receive first grow reply within time-out	relay and set a confirm-timer
14	receive subsequent replies	ignore
15*	receive grow request	ignore

*This transition applies only to the direct grow scheme.

Figure 13. Corresponding transition table.

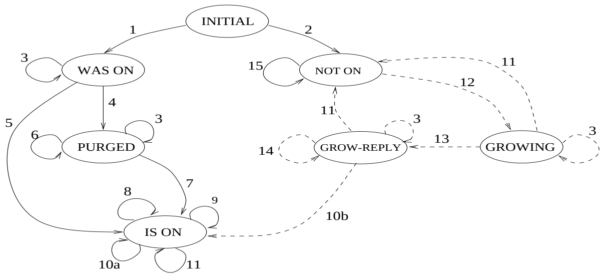

Figure 12. The state diagram.

stage, every switch is in the INITIAL state. If a switch is on the (old) tree, it makes a transition to the WAS ON state. Otherwise, it goes to the NOT ON state. If a switch in the NOT ON state receives a grow request, it just ignores that message and stays at that state.

If a switch in the WAS ON state receives a request, it simply ignores the request. If it receives a repair message, it goes to the IS ON state (i.e. is on the new multicast forest). On the other hand, if it receives a purge message, it makes a transition to the PURGED state. In the PURGED state, the switch sends a grow request to all its neighbors and waits for a reply (note that if the purged switch is MI, it may send a purge message to all its downstream switches except one). If the purged switch receives the first reply from a neighbor, it goes to the IS ON state and becomes a switch on the multicast forest. If it does not get any reply after time-out, it stays in the PURGED state and may try again with a longer time-out value.

If the indirect grow scheme is used, everything described so far applies except the transition from the NOT ON state. More specifically, if a switch in the NOT ON state receives a first grow request message, it now makes a transition to the GROWING state, where the grow request received first will be broadcasted and all subsequent grow requests ignored. As soon as the switch receives the first grow reply (for itself) before a time-out from its neighbor, it relays the grow reply, and makes a transition to the GROW-REPLY state. If there is no confirm message in the GROW-REPLY state after a time-out, it will go back to the NOT ON state (and discard all transition information). If the switch receives a grow-confirm message (before a time-out), it becomes a switch on the new multicast forest and makes a transition to the IS ON state.

When a switch in the IS ON state receives a grow request from say switch X, the switch sends a reply and waits for a confirm message if it is MC or a leaf MI. If the switch is a non-leaf MI, it floods the request, waits for a reply to X, relays the reply to X, and waits for a confirm message. If the switch receives the confirm message, it updates WDM forwarding cache for a new LSP, and if (and only if) it is a non-leaf MI, relay the confirm message.

5. Conclusions

In this paper, we have discussed multicasting in IP over WDM networks. In order to support multicast applications with bursty traffic and/or requiring relatively low bandwidth more efficiently, WDM multicasting based on optical burst/label switching (OBS/OLS) has been proposed as an alternative to IP multicasting (based on WDM unicasting) or WDM multicasting based on wavelength routing. In particular, we have designed a distributed protocol, which modifies a multicast tree constructed by DVMRP (a IP multicast protocol) into a multicast forest where none of the multicast-incapable (MI) switches is used as a branching point.

The proposed protocol has several important features such as using only local information of the WDM layer and not requiring any changes made to the IP multicasting protocol. We have shown that even though some downstream switches of a MI switch may be temporarily removed from the original multicast tree, they can always establish alternate label switched paths (LSPs) and thus be on the new multicast forest to receive multicast data from the source. We have also developed two schemes (the direct and indirect grow schemes) for constructing an efficient multicast forest.

Note that the framework may also be applied to WDM multicasting based on wavelength routing where, instead of LSPs, lightpaths will be established. This work also sheds light on the design of new WDM-layer-aware IP multicasting protocols for the next generation Optical Internet. As a future work, we will evaluate the effectiveness of the proposed protocol in terms of bandwidth saving and latency reduction due to WDM multicasting. Based on the results reported in [5], a huge improvement over IP multicasting (based on WDM unicasting) is expected.

References

[1] C. Brackett, “Dense Wavelength Division Multiplexing Networks: Principles and applications.,” IEEE Journal of Selected Areas in Communications, pages 373-380, Aug. 1990.

[2] P. Green, Fiber-Optic Communication Networks, Prentice-hall, 1992

[3] R. Ramaswami, “Multi-wavelength Lightwave Networks for Computer Communication.,” IEEE Communications Magazine 31, pages 78-88, 1993.

[4] T. Pusateri, “DVMRP version 3,” draft-ietf-idmr-dvmrp-v3-07, obsoletes RFC 1075, August 1998.

[5] R. Malli, X. Zhang and C. Qiao, “Benefits of Multicasting in All-optical Networks,” Proc. of SPIE, All-optical Networking, Vol. 3531, pp. 209-220, Nov. 1998.

[6] C. Qiao and M. Yoo,“Optical Burst Switching (OBS) - A New Paradigm for an Optical Internet, ” Journal of High Speed Networks, vol. 8, no. 1, pages 69-84, 1999.

[7] Bill St. Arnaud, “Architectural and Engineering Issues for Building an Optical Internet,” Proc. of SPIE, All-optical Networking, Vol. 3531, pp. 358-377, Nov. 1998.

[8] G. Sahin and M. Azizoglu, “Multicasting Routing and Wavelength Assignment in Wide-Area Networks,” Proc. of SPIE, All-optical Networking, Vol. 3531, pp. 196-208, Nov. 1998.

[9] L. H. Sahasrabuddle and B. Mukherjee, “ Light-Trees: Optical Multicasting for improved performance in Wavelength-Routed Networks,” IEEE Communications Magazine, Vol. 37, No. 2, pages 67-73, Feb. 1999.

[10] R. Callon, P. Doolan and N. Feldman et. al, “ A framework for Multiprotocol Label Switching,” IETF draft, draft-ietf-mpls-framework-02.txt, Nov. 1997.

[11] E. C. Rosen, A. Viswanathan and R. Callon, “Multiprotocol Label Switching Architecture,” IETF Draft, draft-ietf-mpls-arch-02.txt, July 1998.

[12] J. Moy, “Multicast Extensions to OSPF,” RFC 1584, March 1994.

[13] Xijun Zhang, John Wei and Chunming Qiao, “On Fundamental Issues in IP Over WDM Multicasting,” to appear in Int’l Conf on Comp. Comm. and Networks (IC3N), Oct. 1999.