Fault Tolerant Multiwavelength Optical Rings with Limited Wavelength Conversion[*]

Ori Gerstel (corresponding author)
IBM T. J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, NY 10532,
ori@watson.ibm.com, Phone: (914)784-7193, Fax: (914)784-6225

Rajiv Ramaswami
IBM T. J. Watson Research Center, 30 Saw Mill River Road, Hawthorne, NY 10532,
rajiv@watson.ibm.com, Phone: (914)784-7356, Fax: (914)784-6225

and

Galen H. Sasaki
Dept. of Electrical Eng., University of Hawaii, Honolulu, HI 96822,
sasaki@spectra.eng.hawaii.edu, Phone: (808)956-6103, Fax: 808-956-3427.

Abstract

This paper presents methods for recovering from channel failures, link failures, and node failures in a wavelength-division multiplexed (WDM) ring network, with limited wavelength conversion capabilities at the nodes. Different recovery schemes are presented to handle each type of failure. Each scheme is evaluated based on the network hardware configuration required to support it, and the performance and management overheads associated with fault-recovery.

Keywords: Optical networks, fault-tolerance, self-healing rings.

[*]This work was supported in part by grant MDA 972-95-C-0001 from ARPA.

1 Introduction

Today's telecommunications infrastructure is based on SONET/SDH self-healing rings. To increase the bandwidth of their trunks, the carriers are actively deploying wavelength-division-multiplexed (WDM) point-to-point links. It is likely that WDM ring networks will come next, as a natural evolution of WDM links and SONET rings. A WDM network provides lightpaths. A *lightpath* is an end-to-end connection, comprising of channels interconnected by switches within the nodes with an appropriate configuration. A ring is a popular interconnection topology because it is simple and provides a fair degree of fault-tolerance.

All-optical ring nodes that perform add/drop functions are being actively researched today. It is likely however, that the first practical WDM ring networks will use electronics rather than optics to perform the switching functions within the nodes (O-E-O architecture). Whatever be the architecture of a node, providing limited wavelength conversion capabilities is more cost-effective than full wavelength conversion. This will be quantified later in the paper. Moreover, it has been shown in [1] that if the set of lightpath requests is given in advance, very limited conversion is as good as full conversion.

Failure restoration is particularly important in these networks where lightpaths carry data at very high bit-rates. A customer cannot have a lightpath fail due to a fiber cut, power outage or a component failure. It is also desirable that these failures be handled within the optical network, rather than have the higher layers deal with them, for the following reasons: If the optical network handles the recovery process, it can share the same set of failure-restoration resources among many higher-level networks. This point is further clarified in Section 2.1 where it is shown how many working channels can share a single protection channel. Also, the lightpaths provided by the optical network can support a variety of different protocols, and some of these protocols may not have their own fault-recovery mechanisms.[1]

This work proposes various failure restoration schemes tailored for WDM ring networks with limited wavelength conversion. These schemes handle channel faults, wherein a single channel fails, link faults, wherein an entire link fails, and node faults, wherein a network node fails. We also propose an integrated solution that can isolate and handle all these faults. As will be seen later these mechanisms are very different from those employed in SONET rings, mainly because WDM nodes are based on space switching while SONET is based on time switching. They are also very different from mechanisms for WDM rings with full or no conversion, since such networks allow for much simpler wavelength allocation schemes.

[1]For example WDM links are used to support high-speed mainframe connections between sites [2], and these do not have their own recovery mechanisms.

This paper is structured as follows. Section 1.1 describes related work. Section 1.2 describes the network model and different node architectures. Section 1.3 identifies the different failure conditions. Section 2 proposes and analyzes different recovery schemes. In Section 3, we discuss fault-management aspects of these solutions, and conclude in Section 4.

1.1 Related Work

Extensive work has been performed on fault-tolerant network architectures in general (e.g., [3]) and self-healing rings in particular (e.g., [4, 5]). However, most networks that have such fault-tolerant mechanisms built into them use time division multiplexing on their fibers (SONET, ATM, FDDI are a few of the many examples), and thus can perform the recovery in the time domain. In contrast, WDM networks treat their channels as transparent, continuous streams, and can thus manipulate them only via space switching. This fact, combined with the careful design needed for limited wavelength conversion, substantially complicates the configurations for supporting fault-tolerance. An example for this phenomenon is the support for link versus node failures. While time-based networks typically do not have to distinguish between these cases, WDM rings need to deploy different schemes (see Section 3.2).

Some simple failure restoration techniques for WDM mesh networks have been proposed in [6, 7, 8]. In [6, 7], the idea is to set up an alternate link-disjoint route for each lightpath at the time it is set up. We will see in Section 2 that this method is inefficient in utilizing the available wavelengths. The work in [8] proposes to have a set of protection fiber links deployed in the form of a spanning tree that can be used to provide an alternate route for the lightpaths on a link in the event of the link failing. This method is useful in mesh networks but not very efficient in ring networks.

Several recent papers have considered limited wavelength conversion [9, 10, 11, 12, 1] from a routing and wavelength assignment perspective but none of these consider failure restoration. We believe that our work is the first one to explore different recovery procedures in detail for rings with limited conversion.

Our network model is based on [1] in which its power is demonstrated by configurations that require very limited conversion, yet enables as flexible a routing as networks that employ full conversion. While [1] suggests specific wavelength allocation schemes for allocating wavelengths to a set of given lightpaths, the current work can be coupled with any other wavelength allocation scheme (for example a dynamic wavelength allocation scheme) to provide fault-tolerance in the network.

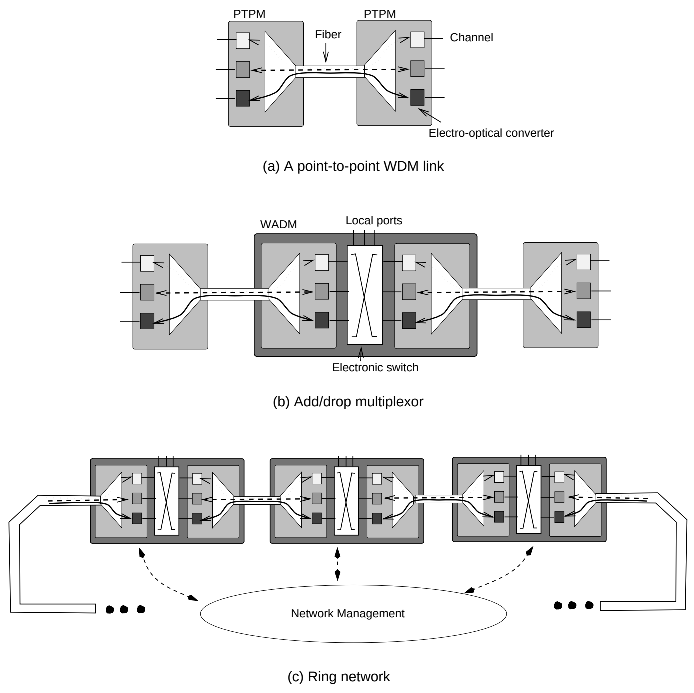


Figure 1: Evolution from point-to-point links to ring networks

1.2 The Model

A WDM point-to-point link, shown in Figure 1(a), uses multiplexors/demultiplexors (abbreviated herein to PTPMs) to combine multiple channels at different wavelengths into a single fiber and to separate the channels at the other end. It is relatively simple to combine such devices into WDM add/drop multiplexors (WADMs): Take a pair of such PTPMs, connect their electronic interfaces to each other, add some switching and local ports and the result is a WADM (see Figure 1(b)). A WDM ring network is then obtained by interconnecting several such WADMs in a ring topology (see Figure 1(c)).

This architecture is based on electronic switching at the nodes (O-E-O), and as a result requires each node to electrically receive each optical channel at each hop, and regenerate an optical signal before multiplexing it into the next fiber. Other architectures are also possible, and are shown in Figure 2. These all-optical architectures differ in the number of wavelength converters at each node and in the tuning capabilities of the converters. Architecture (b) in Figure 2 requires a small number of converters (one per channel); these converters have to be tuned depending on the switching configuration. Architecture (c) in Figure 2 requires a

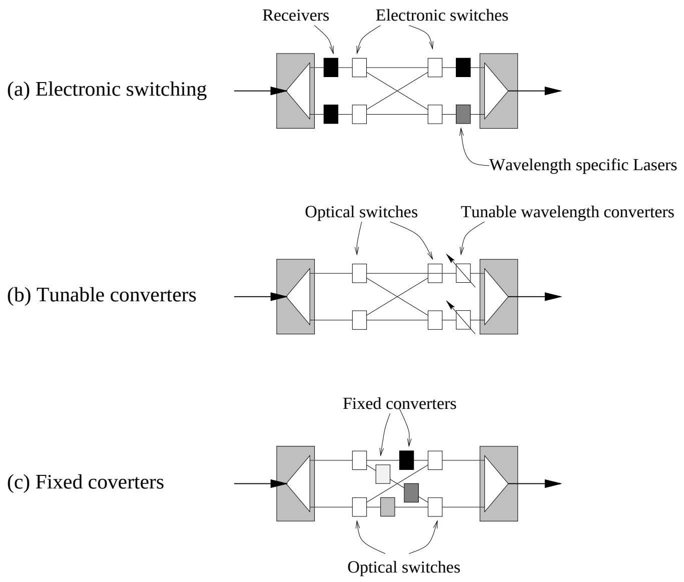


Figure 2: Different architectures for a WADM

converter per connection between switches (i.e., if there are W channels each with a switch of degree Δ, this solution will require $W\Delta$ converters). However these converters need no tuning. While these architectures are very similar in their fault-tolerance schemes (in particular option (a) and (b)) and most of the paper can be easily modified to fit the specific architecture, there are several differences. For the sake of simplicity we focus on option (a) exclusively in what follows.

Within each WADM, different types of wavelength conversion are possible, each requiring different amounts of switching and conversion capability, as shown in Figure 3. *Fixed* conversion does not require any switches, as shown in Figure 3(a). The case of *no* conversion is just a specific form of this interconnection pattern but this does not utilize the wavelengths efficiently [13].

On the other hand, one could have a full blown non-blocking switch/conversion between the channels (as in Figure 3(b)). Such a solution enables any-to-any conversion in each and every node. It also allows for simple switch management and simple, yet optimal, wavelength allocation. However such a node is likely to be very expensive to realize. An intermediate solution is to install switches of small-degree for each channel, and to connect them to the switches of other channels in some clever fashion (see Figure 3(c)). This solution is far less expensive than the full conversion case (depending on the degree Δ of the the switches), yet supports many of the scenarios that full conversion supports [1], provided that the interconnection pattern is

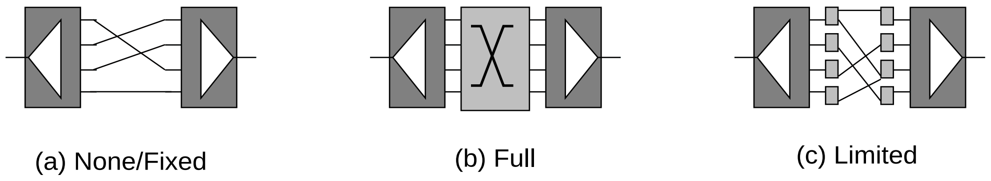


Figure 3: The spectrum of wavelength conversion capabilities in a WADM

carefully planned. However it does add some complexity to the failure restoration mechanisms required. As we will see, careful planning enables the network to offer the same level of fault-tolerance that full conversion supports.

1.3 Failure Scenarios

We identify the following three main types of failure scenarios:

Channel fault: In this case, a single channel on a link between two nodes has failed. This failure can result from a failure of the designated laser or receiver for the channel, or wire disconnections. This failure should be dealt with locally, by routing the traffic from the faulty channel to a spare channel on the same physical link. We present solutions that can handle multiple channel faults. Our solutions can be used for point-to-point links as well.

Link fault: This is typically caused by a fiber cut. Such failure can be dealt with by using a separate protection fiber, or by providing a "loopback" mechanism within each node on the same working fiber, without using a separate protection fiber. We describe these solutions in the next section.

Node fault: In this case, an entire WADM node has failed. This is the most severe fault of the three, and may result from power outages or catastrophic scenarios. This is also the most complex scenario to deal with and cannot be handled without coordination between the nodes.

Throughout the paper we shall use N to denote the number of nodes in the ring and W to denote the number of wavelengths (channels) per fiber. We shall also use the term *link* to denote the transmission medium connecting adjacent nodes (typically a single fiber/fiber pair, with an additional fiber/fiber pair for protection if desired). The term *channel* is used to denote the logical link between nodes that uses a specific wavelength. Thus W channels are multiplexed onto each of the N links comprising the ring. We also assume full duplex links and lightpaths. Such topologies are typically realized by pair of counter-propagating resources, and are much easier to manage.

2 Fault Tolerant Configurations

In this section we present WADM node configurations that deal with the fault scenarios mentioned above. In all cases, cost considerations dictate that the fault-tolerance support require as few additional switches as possible, or even better, only a few additional ports in existing switches. This overhead will be referred to as the *switching overhead*. Another important design factor is to reduce the number of switching elements involved in the data path, since this reduces the deterioration in signal quality. The maximum number of switching stages in a node will be referred to as the *hop overhead*. A third crucial factor, which can dominate the reaction time from detection of a fault to restoration of service, is the need to coordinate between nodes in the ring (or between the nodes and the network management site). This will be referred to as the *coordination overhead*.

2.1 Channel Faults

The least cost solution for backing up channels for the case of channels failures is to allocate one channel on each link between WADMs for backup purposes. Upon detection of a channel failure, the switches in the node switch the data from the failed channel to the backup channel. This solution can be extended to support multiple channel failures.

We propose three solutions, one requiring very small switching overhead (two ports per channel are used for fault-tolerance) and very low hop overhead (a maximum of two hops per channel), but with a big drawback: in order to move from the normal configuration to the backup configuration, the node needs to be fully reconfigured, resulting in short disruptions to all the lightpaths using the link. A second solution, which does not disrupt lightpaths not using the failed channel, has much larger switching overhead and hop overhead. Both solutions do not have a coordination overhead and allow $W:1$ and $W:2$ channel protection. These solutions are primarily useful for point-to-point links (i.e., in each PTPM). A third solution that can be integrated with the mechanism to handle link failures is proposed in Section 2.2.

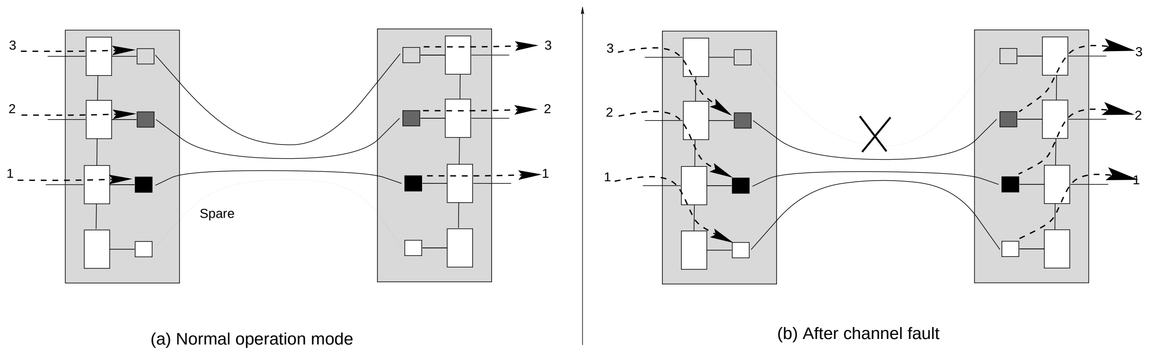


Figure 4: Protection against a single channel failure (solution 1)

2.1.1 Solution 1: Allowing Full Reconfiguration

Logically arrange the channels in each PTPM in a chain. Use two ports of the switch of each channel to connect the switch to the switch of the previous and next channels in the chain (see Figure 4(a)). Let channel 0 be the designated backup channel. Upon failure of some channel i, configure the switch of i to transfer i's data to channel $i-1$ on the link. Also configure the switch of channel $i-1$ to transfer its data to channel $i-2$. Repeat the process until channel 1's data is transmitted on channel 0 (see Figure 4(b)).

The same process is repeated in the PTPM on the other side of the link upon detection of a fault. As a result, after a short disconnection, many of the lightpaths use different channels than the original ones (the exact number being i).

This solution can be readily extended to support two channel failures: Designate channel $W-1$ as another backup channel. Now, upon the first failure (of, say, channel i), reroute connections $i, i-1, ..., 1$ to $i-1, i-2, ..., 0$, as before. Upon a second failure of channel j, reroute j to $j+1$, $j+1$ to $j+2$ and so on, until channel $W-2$ is rerouted to the spare channel $W-1$ (see Figure 5).

This configuration suffers from a large number of temporary disconnections for transforming from normal to fault configuration and vice versa. An improved configuration connects the switches in a ring rather than a chain. It is then possible to choose the shorter path from the failed channel to backup, resulting in a reduction of 50% in the number of channels that need to be rerouted (on the average).

In most cases, two backup channels should suffice. Larger numbers of channel faults can be dealt with more efficiently by the link fault solutions of Section 2.2. In any case, the generalized configuration for k spare channels should provide k node disjoint paths from any set of k nodes to the set of spares. Note that

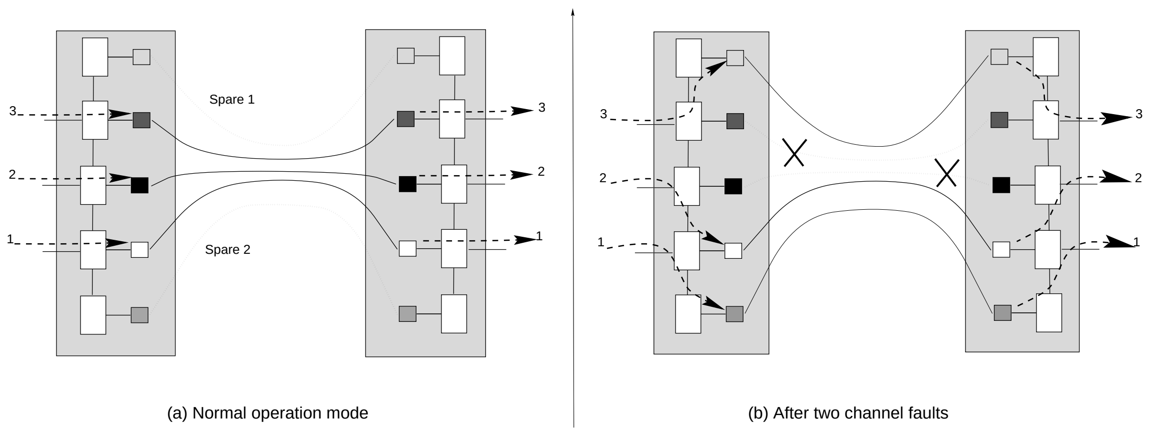


Figure 5: Protection against two channel failures (solution 1)

there is no requirement for short paths to the spares (since no signal actually traverses the path). Also note that the switches used in this case can be blocking switches, that is, they need only support one connection through them (if the switch currently connects input port i to output port j it cannot connect any other input port to any other output port).

Overheads:

Switching: Two additional ports per switch are required. These switches can be blocking.

Hops: Each channel goes through at most two switches in a node: its own and that of the adjacent channel.

Coordination: Since both ends of the link perform identical operations, no coordination is needed.

2.1.2 Solution 2: No Unnecessary Disconnections

To support a single failure, arrange the channels within each PTPM in a tree structure as shown in Figure 6(a). The root of the tree is the switch of the backup channel b, and its degree, $\Delta > 2$, depends on the number of ports per switch that can be allocated for fault-tolerance purposes.

During normal operation, each switch transfers data from its I/O port to the link. When a channel i fails, the switches on the path from i to b are configured to transfer i's data to b. (See Figure 6(b) in which the switch of the faulty channel is marked "F". Note that it is assumed that the optics has failed but the switch itself has not failed.) This operation must be done without disrupting the transfer of data on the

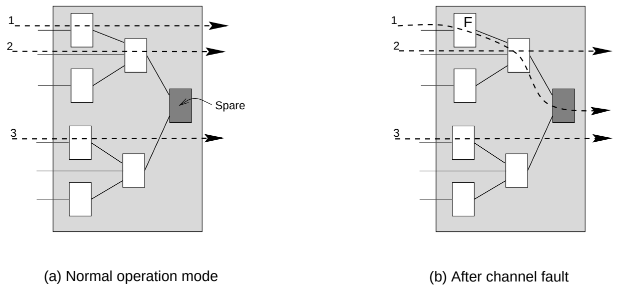


Figure 6: Protection against a single channel failure (solution 2)

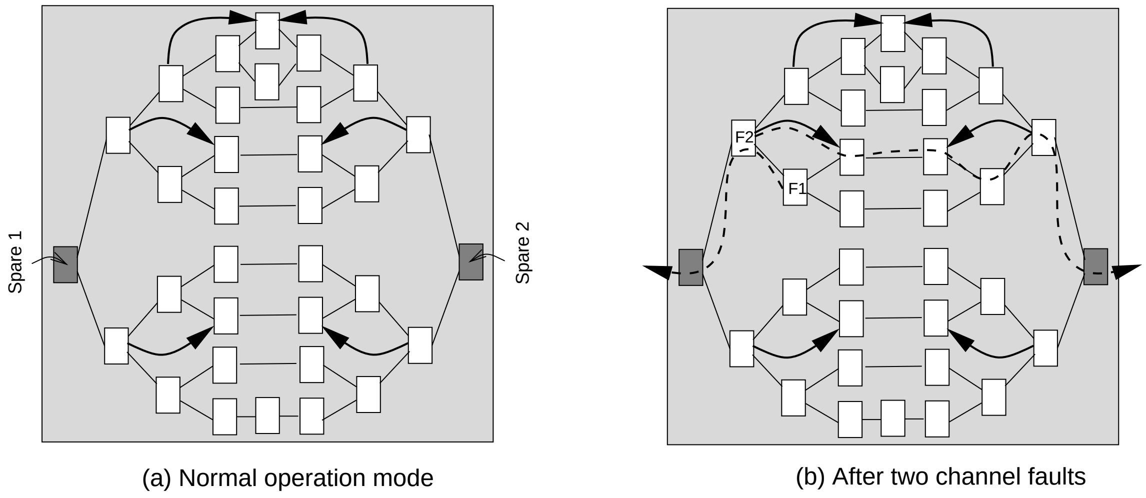


Figure 7: Protection against two channel failures in a 33 channel system (solution 2)

other channels from their I/O ports to the link. This requires the switch for each channel to be non-blocking. Clearly, the maximum length of the path from a node to the root is $\log_{\Delta-1} W$.

This case may again be generalized for two backup channels as follows. Arrange the switches in two trees, each tree rooted at a switch of a backup channel. These trees are connected at the leaf level, to form a *dual tree* (see Figure 7). An additional port (termed *bypass* herein) is allocated in each channel's switch to connect the switch to a switch of a leaf node (thick arrows in the figure). When a failure occurs, the data of the failing channel is routed via the bypass port to a leaf, and from there, through either of the trees to the backup channel. When the second failure occurs, the second tree is used to route the data to the second backup channel.

To enable this connection pattern, there should be a sufficient number of free ports in the leaves to support all bypass connections from the other tree nodes. This is guaranteed by the following lemma, the proof of which is given in the Appendix.

Lemma 1. *Assume the degree of each node is $\Delta \geq 4$. Then the number of free ports in the leaves of the dual tree is larger than the number of internal nodes in it.*

As in the previous section, two spare channels are enough for practical purposes. The configuration for the general k spare case should satisfy the following conditions. (i) For any set of k failed channels, there should be k edge disjoint paths to the spares, (ii) these paths should be short ($O(\log_\Delta W)$ seems feasible), and (iii) the configuration should be non-blocking.

Overheads:

Switching: Given Δ ports are allocated for protection purposes, $\Delta > 2$ s required for single channel protection and $\Delta > 3$ for supporting up to two failures. Note that the switches have to be non-blocking (as opposed to the previous solution).

Hops: Each channel goes through at most $2 + \log_\Delta \frac{W}{2}$ switches in a node since one hop leads to a leaf, from which yet another hop may be necessary to arrive at the leaf of a full Δ degree tree containing half of the channels, leading to the currently unused spare.

Coordination: Since both ends of a link perform identical operations, no coordination is needed.

This scheme works well for certain numbers of channels (W). A more detailed scheme that optimizes the design for any value of W is given in the appendix.

2.2 Link Faults

There are four common techniques to protect against link faults:

Span protection: Add a protection fiber between each pair of adjacent nodes, which will be used by the nodes when the primary fiber has failed. Such a recovery scheme is deployed in SONET BSHR/4 rings[2].

[2] Bidirectional SONET rings that use four fibers per link between adjacent nodes, two unidirectional working fibers in opposite directions and two protection fibers. Such rings are also called BLSRs.

Line protection: Mend the lightpaths affected by the fault locally, by replacing the faulty link by a loop around the ring on the working fibers but on separate backup wavelengths (see BSHR/2 rings[3] for a similar scheme).

Online path protection: When a link fails, set new routes around the ring from the sources of affected lightpaths to their destinations, and reroute the lightpaths to the new routes.

Proactive path protection: For each lightpath, set two routes in advance, on the two alternatives around the ring. When one such route fails — switch to the other (e.g., USHR/P rings[4] [6, 7]).

As mentioned earlier, despite the conceptual similarity of these schemes to TDM protection schemes, the implementation is quite different due to the fact that space switching is used.

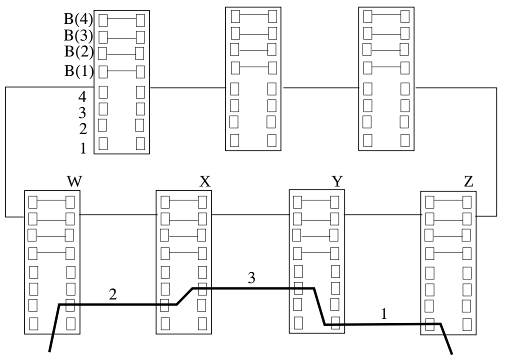
Online path protection may not be a viable option because the complexity of coordinating the endpoints of a lightpath to find an available backup route around the ring may take hundreds of milliseconds, which is not likely to be tolerated by many higher level protocols. Proactive path protection is not a reasonable solution either as it is inefficient in utilizing the available wavelengths. With W wavelengths available, one can only support a total of W simultaneous lightpaths. Moreover if all W happen to be single-hop lightpaths, then we end up using $(N-1)W$ channels for failure restoration and only W channels for carrying the actual lightpath, a very poor channel utilization ratio. (Recall that a channel corresponds to a wavelength on a given link.) With line protection we can support many more than W simultaneous lightpaths. In the best case, if all lightpaths are single-hop lightpaths, then our line protection scheme allows us to support $N\frac{W}{2}$ simultaneous lightpaths.

Span protection is a very attractive solution, as it is very simple, fast, and can be done locally at the two ends of the failed link. All that is necessary is to add an optical switch in the node's output, just in front of the fiber, and to control which fiber is to be used based on detection of light on both fibers. However, this solution is not viable if the cost of additional fiber is high.

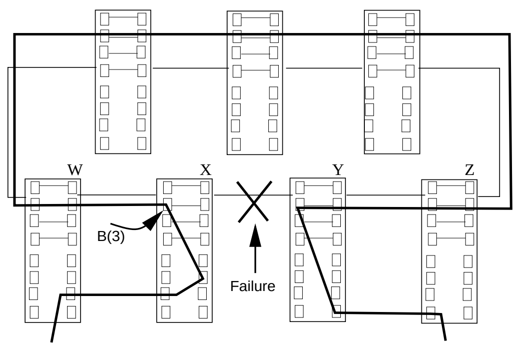
The advantage of line protection over proactive path protection is that it requires only up to 50% of the channels in the network. This is clearly the minimum required in any scheme if all lightpaths must be protected. Its advantage over span protection depends on the relative cost of the additional 50% transmitter/receiver pairs versus the cost of additional fiber. Another advantage is that while in the case of span

[3] Bidirectional SONET rings that use two opposite unidirectional fibers per link between adjacent nodes, each of which is utilized up to half of its capacity.

[4] Unidirectional SONET rings employing two opposite fibers, and transmitting the data on both of them simultaneously. Also termed UPSRs.



(a) Before failure



(b) Connection restored after failure

Figure 8: Line protection

protection the backup fiber is idle at normal operation, the extra transmitter/receiver pairs needed for line protection can be used for low priority connections (which are taken down when the backup channels need to be used for fault-recovery).

In what follows we focus on the line protection mechanism. This mechanism is based on allocating up to half of the wavelengths for protection, with each data channel i having its unique backup channel $B(i)$. When a failure occurs at a link, it is detected by the nodes adjacent to the failing link, which configure their switches to loop back all the lightpaths that use the link (see Figure 9(b)), through the rest of the ring all the way to its other end, where they are switched to the remaining part of their route. Since backup channels have a default configuration that enables a signal to travel around the ring ($B(i)$ of one link is configured by default to connect to $B(i)$ on the other link in each node, the operation can be completed locally at the nodes that discover the fault.

The intricate part of the design pertains to the actual loopback mechanism at the nodes. Take for example the lightpath in Figure 8(a). It uses wavelength 2 on link WX, wavelength 3 on XY and wavelength 1 on YZ. Suppose XY fails. After the failure, this lightpath must be looped around the ring via XW...ZY and converted from wavelength 2 (on link WX) to wavelength 1 (on YZ). We propose the configuration shown in Figure 9(a), which requires only a single additional port within each switch. Going back to the example in Figure 8(a), connect the switch of channel 3 on link XY to the backup channel $B(3)$ on link WX, and activate this connection upon failure. The lightpath will now use channel 2 on link WX, get converted to 3 in node X, from which it will be converted again to $B(3)$, loop around the ring on channel $B(3)$ until it

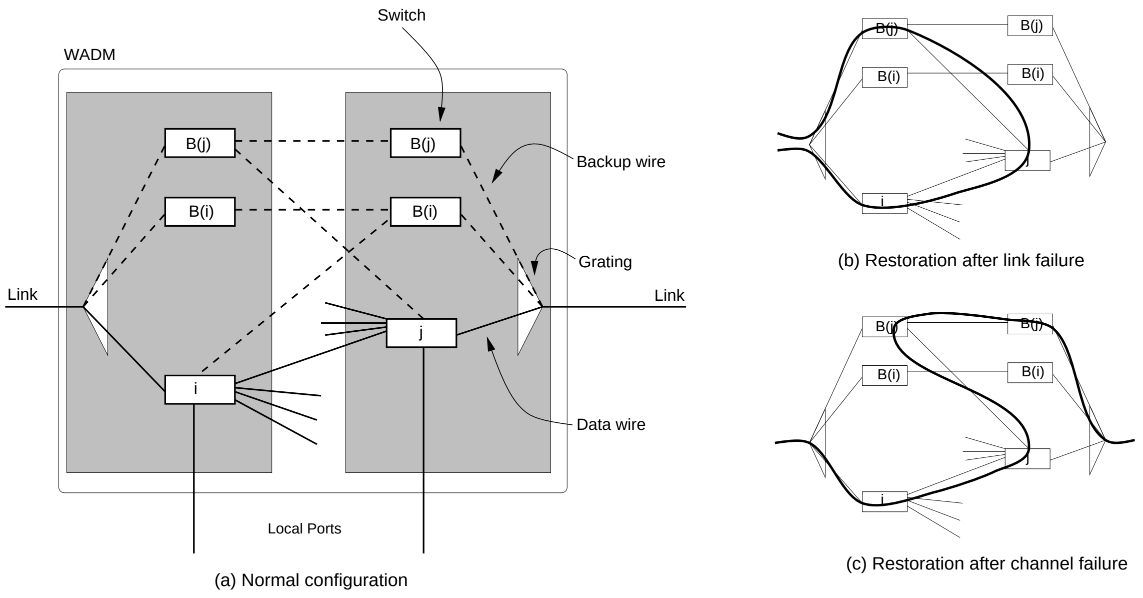


Figure 9: WADM node configuration supporting line protection

reaches node Y. At node Y it is converted to 3, and converted again to 1 on link YZ (see Figure 8(b)).

Overheads: The overheads involved are as follows:

Switching: One additional port per switch is required.

Hops: Each channel goes through at most three switches in a node. However, a looped-back lightpath goes through N additional nodes.

Coordination: Since both ends of a link perform identical operations, no coordination is needed.

Observation 1. *This scheme can handle single channel failures as well. Consider the configuration in Figure 9(c). Using this configuration if channel j fails, the data on it can be switched to channel $B(j)$ on the same link. This type of restoration will require a management protocol to prevent a link failure event and a channel failure event from simultaneously triggering an attempt to use the same backup channel for recovery. This issue will be addressed in Section 3.*

Observation 2. *The channel failure restoration scheme described above does not allow channel i on* both *links of a WADM to be restored in the event of both channels failing. This can be rectified by adding an* *additional connection between the $B(i)$ switches.*

The amount of protection against channel failures in this case is maximized: it is possible to simultaneously recover all working channels in all the links around the ring.

2.3 Node faults

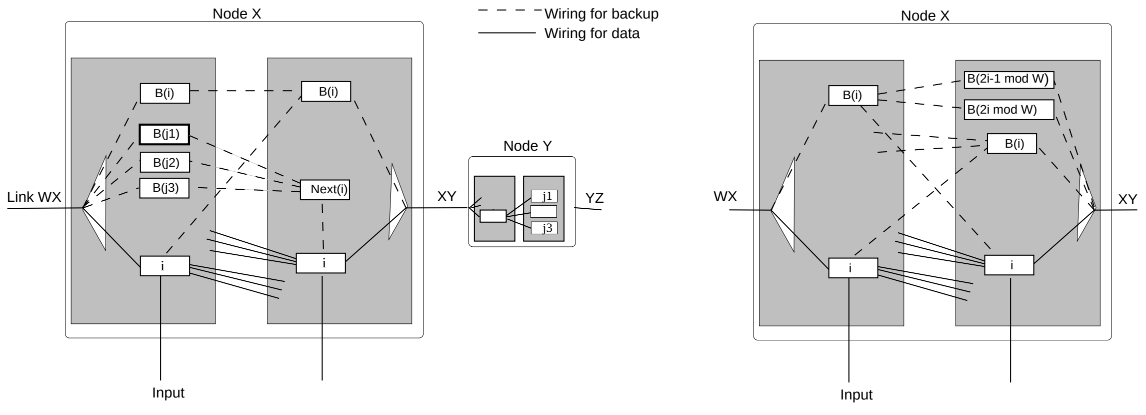
Node failures are complicated to handle, because part of the conversion capability of the network as a whole is lost when a node fails. The rest of the network must compensate for this loss by incorporating more switching capability to support the failure. Note that in the other cases, we assumed that all the switches are still available to handle the restoration procedures.

We present two solutions for supporting node failures. The first solution requires 25% more switching hardware but enables the reconfiguration to be performed with no coordination between nodes. This solution is an extension of the line protection scheme proposed in Section 2.2. The second solution is based on a different approach. It does not require more switching hardware, but requires global coordination in the network.

2.3.1 Solution 1: More Hardware

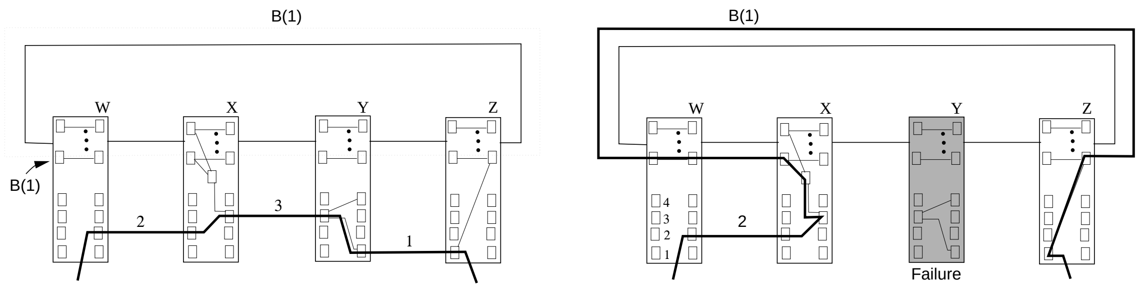
In order to enable a solution along the lines of the link failure solution, one must overcome the following obstacle. Assume channel i supporting lightpath p has failed on the link between nodes X and Y. In the link failure case it is possible to convert to a backup wavelength, $B(i)$, at node X and to use $B(i)$ to loop around the ring, arrive at node Y, and then convert back to i in order to continue along the original course of the lightpath. In contrast, suppose that node X, converting p from wavelength i to wavelength j, fails. Then there is no suitable channel to convert p to at both nodes adjacent to the failure (neither $B(i)$ nor $B(j)$ can be chosen without adding more wiring at one of the neighbors).

Number the nodes from 0 to $N-1$ clockwise around the ring and call $(i+1) \bmod N$ the clockwise neighbor of i. Our scheme adds more conversion capability to the clockwise neighbor of a failing node, while the other neighbor performs the same operation as for the link fault case. Referring to Figure 10(a), the following additional wiring is necessary at node X: if node Y can convert wavelength i on link XY to wavelengths $\{j_1, .., j_k\}$ on link YZ, then (at node X) we connect the switch of channel i on link XY to an extra switch, $Next(i)$, which is connected back to channels $\{B(j_1), ..., B(j_k)\}$ on link WX. As in the link failure case, at



(a) Node configuration for solution 1 (b) Node configuration for solution 2

Figure 10: Node configurations to support recovery from node failures



(a) Before failure (b) Connection restored after failure

Figure 11: Protection against node failures (solution 1)

node X we also connect channel i on link WX to channel $B(i)$ on link XY and similarly channel i on link XY to channel $B(i)$ on link WX.

Suppose now that node Y fails, as shown in Figure 11(b). Then its clockwise neighbor X converts the affected lightpath p shown in the figure to $B(1)$, where 1 is the wavelength used by p between the failed node Y and its other neighbor Z. At the same time, node Z converts from 1 to $B(1)$ as well, to re-establish the lightpath. Note that node Z performs the same function for the failure of node Y as for the failure of link YZ. However node X performs a different function depending on whether node Y has failed or whether link XY has failed (see Section 2.2). Distinguishing between these two cases is part of the management function to be discussed in Section 3.

Overheads: The overheads involved in this configuration are the following:

Switching: $\frac{W}{2}$ *Next*(i) additional switches per node are required with a degree of Δ each (where Δ is the degree of a a switch without the protection mechanism). This represents an increase of 25% in the number of switches since there are $2W$ regular switches per WADM. As a result, the $\frac{W}{2}$ backup switches connected to the *Next*(i) switches need Δ more ports each. All other switches need a single extra port (i.e., their degree in $\Delta + 1$).

Hops: Each channel goes through at most four switches in a node.

Coordination: Assuming that a node failure diagnosed, this scheme can be performed with no additional coordination, as follows. Upon setup of a lightpath, each node records the conversion of the lightpath in the *next* clockwise neighbor. Returning to the example in Figure 11(a), node X records that the lightpath using wavelength 3 on link XY is converted to wavelength 1 at node Y. This can be elegantly done by setting the switch *Next*(*3*) at node X to $B(1)$. Upon failure, node X sets its switch of channel 3 on link XY to *Next*(*3*) (rather than to the grating), and node Z sets its switch of channel 1 to $B(1)$ — both of which actions are based entirely on internal node information.

2.3.2 Solution 2: Global Coordination

The next solution requires no additional hardware (except for an additional port per switch), and the lost conversion capability due to a node failure is compensated for by having a richer connectivity pattern between the switches of backup channels.

When a node Y fails, each lightpath x_i that goes through Y uses some channel $Left_i$ to the left of Y and some channel $Right_i$ to the right of Y. For the sake of simplicity assume that all $\frac{W}{2}$ possible lightpaths go through Y (all of the wavelengths are used and none of them is dropped/added). Thus, the pairs $\{(Left_i, Right_i)\}_{i=1}^{W/2}$ represent a *permutation* of the values $\{1, ..., \frac{W}{2}\}$. If each neighboring node sends a failed channel i to its backup $B(i)$ — as in the link failure case — and the set of backup channels is configured as a $\frac{W}{2} \times \frac{W}{2}$ permutation network (e.g., a Beneš network [14, 1]), then the backup channels can route each $Left_i$ channel to the correct $Right_i$. Note that, as opposed to the classic use of permutation networks, we do not use it as a switch fabric configuration but distribute it over a whole ring, having a single stage of it in every node, as in [1].

For this case it is important to choose the appropriate permutation network topology. For example, the Beneš network is not a good choice here, since we need a network that functions as a $\frac{W}{2} \times \frac{W}{2}$ switch, no matter where the node failure is. In other words, we need a cyclic configuration around the ring, that can be cut at any point, to yield a $\frac{W}{2} \times \frac{W}{2}$ rearrangeable permutation network. Clearly, if all the stages in a permutation network are identical, its input and output $\frac{W}{2}$ ports may be connected to arrive at the required configuration.

The basic building block for this configuration is the *Shuffle/Exchange* pattern from a set S_1 of switches to a set S_2 (where both sets contain $\frac{W}{2}$ switches). This pattern connects each switch $i \in \{1, ..., \frac{W}{2}\}$ in set S_1 to switches $(2i-1) \bmod \frac{W}{2}$ and $2i \bmod \frac{W}{2}$ in set S_2. The network resulting from concatenating $\log_2 \frac{W}{2}$ stages of Shuffle/Exchange in a row is called an *Omega network* [15]. We shall use the following results on Omega networks.

Theorem 1 [15]. *An Omega network is equivalent to a butterfly network.*

Corollary 1. *Two Omega networks connected back-to-back form a full scale rearrangeable permutation network.*

Theorem 2 [16]. *Let a* Triple-Omega network *be the result of concatenating three Omega networks in a row. Then a Triple-Omega network is a full scale rearrangeable permutation network.*

The next result was obtained by an exhaustive search. It is still an open question whether it holds for any value of W.

Theorem 3 [16]. *Let a* $\frac{W}{2} \times \frac{W}{2}$ Double-Omega network *be the result of concatenating two* $\frac{W}{2} \times \frac{W}{2}$ *Omega networks in a row. Then if* $W \leq 16$*, a Double-Omega network is a full scale rearrangeable permutation network.*

Let S_1 be the set of switches of backup channels on the link connecting a given node to its clockwise neighbor, and S_2 the set of switches connecting the node to its other neighbor. The following configurations are appropriate for our purposes:

Config-1. Wire the switches of backup channels in each node according to a Shuffle/Exchange from set S_1 to S_2 (see Figure 12). If the size of the network is large enough, namely: $N \geq 3 \log_2 \frac{W}{2} + 1$, then after a node failure, the remaining backup switches form a Triple-Omega configuration and can route the

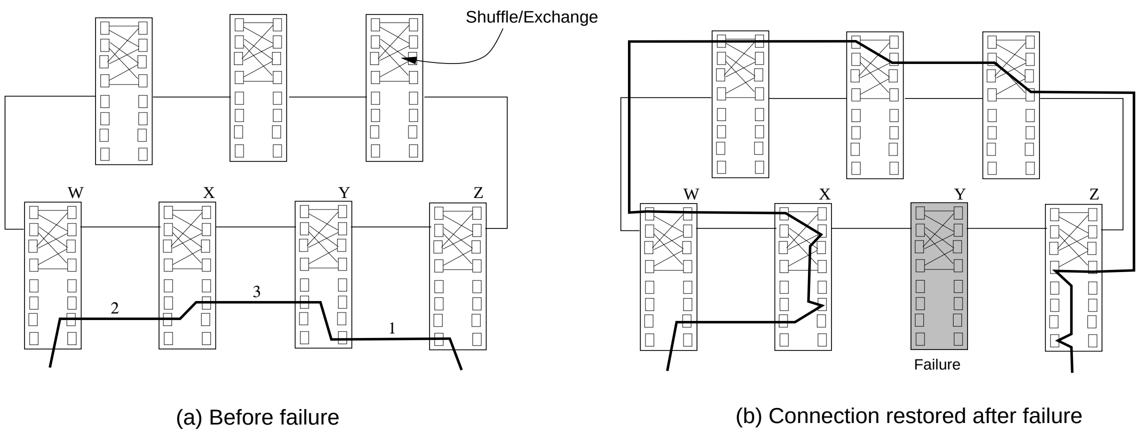


Figure 12: Protection against node failures (solution 2)

broken lightpaths to realize the required permutation[5]. If $W \leq 16$ it is enough to have $N \geq 2\log_2 \frac{W}{2}+1$ nodes.

Config-2. Wire the switches in each node according to a Shuffle/Exchange between set S_1 and S_2. Also wire the switches in a Shuffle/Exchange pattern from set S_2 to S_1. By ignoring half the wiring, a sequence of $\log_2 \frac{W}{2}$ can be used as an Omega network. By ignoring the other half of wires, the same network can be used as a reverse Omega network. Thus $2\log_2 \frac{W}{2}$ stages can be used as a pair of Omega networks connected back to back, which can perform any permutation. This solution requires twice as many ports at each node compared to the previous configuration. However it works for smaller networks with $N \geq 2\log_2 \frac{W}{2} + 1$ and works for any W.

Note that in both schemes, we effectively distribute the permutation network P over the entire ring. This is done by first connecting the ports of one side of P to the ports on the other side, thereby transforming the linear structure of P into a circular structure P'. Then, the number of nodes is doubled by splitting each node at phase i of P' into two parts a and b: part a interfaces with nodes in phase $i-1$ of P' and part b interfaces phase $i+1$ of it. In addition the two parts of the node are connected by a new connection. This new network, P'', clearly retains all the properties of the original P'. Finally, we embed P'' onto the ring by embedding each node a of phase i into an S_1 switch of a channel at node i of the ring, and node b of phase i into an S_2 switch of node $(i+1)$ mod N of the ring. The connection between parts a and b of the same

[5]Note that the Triple-Omega network has only $3\log_2 \frac{W}{2}$ stages. We need one additional stage in our network to realize our desired permutation, since a node failure eliminates one of the stages in the permutation network.

node is mapped onto an optical channel.

Recovery from a node failure is done in a centralized manner by a network management entity. First it has to figure out which connections are affected by the fault. This can be done by storing the current lightpath configuration in the management site and modifying it as requests arrive and depart the system. After the fault it is easy to compute from this configuration the actual permutation between the $(Left_i, Right_i)$ pairs mentioned earlier. At this point, the circular permutation network is a linear network again, and its inputs/outputs are defined by the channels adjacent to the fault from both sides of the ring. On this network the configuration of the backup switches is computed to meet the required permutation. Finally, network management notifies each node of its local configuration.

Overheads: The overheads involved in these configurations are as follows:

Switching: Switches of backup channels need only have four ports in Config-1 (two ports for the Shuffle/Exchange, one for connecting to the grating and one for connecting to the corresponding non-backup channel). In Config-2 each of these switches needs to have six ports. Data channel switches need only one extra port, as for the link failure case.

Hops: Each channel goes through up to three switches in a node. However, a looped-back lightpath goes through N additional nodes.

Coordination: It is necessary to set up the entire configuration of backup switches to support the permutation. Thus all the network nodes have to coordinate in this case.

3 Fault Management

So far we have described the necessary mechanisms in the nodes to restore connections from a failure. Two additional higher level management aspects are still necessary to complete the picture. The first aspect addresses the need to manage single channel failures versus link failures in the case of Section 2.2. The second aspect addresses the need to diagnose the fault type before action is taken. This enables us to integrate the schemes in this paper into one unified channel/link/node recovery scheme.

3.1 Managing channel and link faults

The scheme of Section 2.2 works correctly if a single link fails. The scheme can also handle multiple channel failures on different links. This is better than adding the costly hardware mechanism of Section 2.1 to deal specifically with channel faults. However, if no further steps are taken, channel faults and a link fault can cause collisions, since both of them use the same set of backup channels for protection. This will result in corrupting lightpaths that recovered from channel failures and lightpaths that recovered from a link failure and even more severe data security problems.

Network management must ensure that if $B(i)$ is already used for recovering from a channel fault, it will not be used to recover another lightpath at the same time. The challenge here is to devise a mechanism that does not require global processing at the time of a failure, in order to keep the restoration time as small as possible.

To manage the backup channels it is necessary to maintain local flags at each node which control the decision to use each backup channel for recovery purposes. The solution requires some global management entity (GME) to maintain the flags. More specifically, each node x maintains two boolean flags for each backup channel $B(i)$: $EnableChan[i, x]$ and $EnableLink[i]$. $EnableChan[i, x]$ controls channel recovery using $B(i)$ at x, while $EnableLink[i]$ controls the use of $B(i)$ for link recovery at x. Note, by the notation, that while an instance of $EnableLink[i]$ appears in every node, $EnableLink[i]$ is identical in all the nodes while $EnableChan[i, x]$ may be different. In addition to the nodes themselves, GME has an updated copy of all the flags (for every i and x). All the flags are set only by GME. The algorithm to maintain these flags is given in Figure 13.

An important characteristic of this process is that it does not slow down the actual recovery process, since all decisions are based on local flags only. The slower global coordination by GME that follows is adequate as long as two failures do not occur very close to each other (which is not expected to be the case).

3.2 Integrated recovery mechanism

This section attempts to merge the recovery scheme for channel and link faults presented in Section 2.2 and the *local* recovery scheme for node faults (Section 2.3, Solution 1). As all schemes are local, it is desirable to merge them into a unified scheme that handles all fault scenarios quickly, without coordination. Unfortunately, since these recovery schemes are not identical, it is necessary to diagnose the fault type in order to activate the appropriate scheme, a process that requires some coordination.

1. Upon failure of a single channel i at node x:

    (a) If i is not used by any lightpath at node x, ignore the failure. Otherwise:

    (b) Node x checks $EnableChan[i, x]$. If it is on, x proceeds with channel recovery as in Section 2.2 and informs GME of the event. Otherwise – no recovery is done.

    (c) Upon receipt of a notification, GME turns the $EnableLink[i]$ flag off at all the nodes. It also turns $EnableChan[i, x]$ off.

2. Upon failure of channel i as part of a link failure adjacent to node x:

    (a) If i is not used by any lightpath at node x, ignore the failure. Otherwise:

    (b) Node x checks its local instance of $EnableLink[i]$. If it is on, x proceeds with link recovery as in Section 2.2 and informs GME of the event. Otherwise – no recovery is done.

    (c) Upon receipt of a notification, GME turns $EnableLink[i]$ off at all the nodes. It also turns $EnableChan[i, y]$ off for every node y.

3. Upon restoration of service of channel i at node x:

    (a) If i is using $B(i)$ as part of the channel recovery scheme, it resets the switches to normal mode and informs GME. Otherwise – it ignores the event.

    (b) GME turns $EnableChan[i, x]$ on.

    (c) If $EnableChan[i, y]$ is on for every node y, GME turns $EnableLink[i]$ on.

4. Upon restoration of service of a link adjacent to node x:

    (a) For every channel i that is using $B(i)$ as part of the link recovery scheme at node x, x resets the switches to normal mode and informs GME.

    (b) GME turns $EnableChan[i, y]$ on for every node y. It also turns $EnableLink[i]$ on.

Figure 13: Maintenance of protection resources for unified channel and link recovery

Diagnosing a channel failure from the other types is easy: upon failure of channel i, the node checks if channel $B(i)$ is operational. If so — this is a channel failure and the diagnosis is complete. Otherwise, it is either a link or a node failure. (This can also be detected by determining whether light is present on the other channels on the link or not.)

Handling both link and node failures in networks with limited conversion without first diagnosing the fault type appears to be hard for the following reasons:

- It is not possible to apply only the link recovery scheme since node failures are not supported by it,
- It is not possible to apply only the node recovery scheme since if only a link has failed such a scheme ignores the single hop lightpaths affected by the fault. Also, the next hop node has to be notified that some fault has occurred, as the link fault is not adjacent to it, and
- If link recovery is applied to single hop lightpaths and node recovery to the rest, then the same $B(i)$ may be used by both schemes to recover two different lightpaths.

The following scheme integrates the diagnosis and recovery process and is demonstrated on the generic example in Figure 14.

Node configuration. We adopt the node configuration for node failures (Figure 10(a)) and add to it an additional connection between the switch of channel i on link XY to that of $B(i)$ on link WX (thereby merging into it the wiring scheme for link faults).

Phase 1. The nodes adjacent to a fault *assume* it is a link fault and activate the link recovery procedure (nodes X and Y in Figure 14(a), nodes X and Z in Figure 14(b)).

Phase 2. The clockwise neighbor of a failed link XY (node X) sends a message to the second hop neighbor (node Z), announcing that link XY has failed. This is done via some out-of-band signaling network (e.g., IP) or via some in-band signal propagating anti-clockwise (e.g., on a dedicated supervisory wavelength).

Phase 3. Node Z receives the message and examines to see if link YZ has failed. If not, Z sends a reply to X diagnosing the fault as a link fault. If YZ has indeed failed, this is a failure of node Y, and Z notifies X of a node failure. Note that the switch setting of Z is already correct since the anti-clockwise neighbor of a fault activates the same reconfiguration regardless of the fault type.

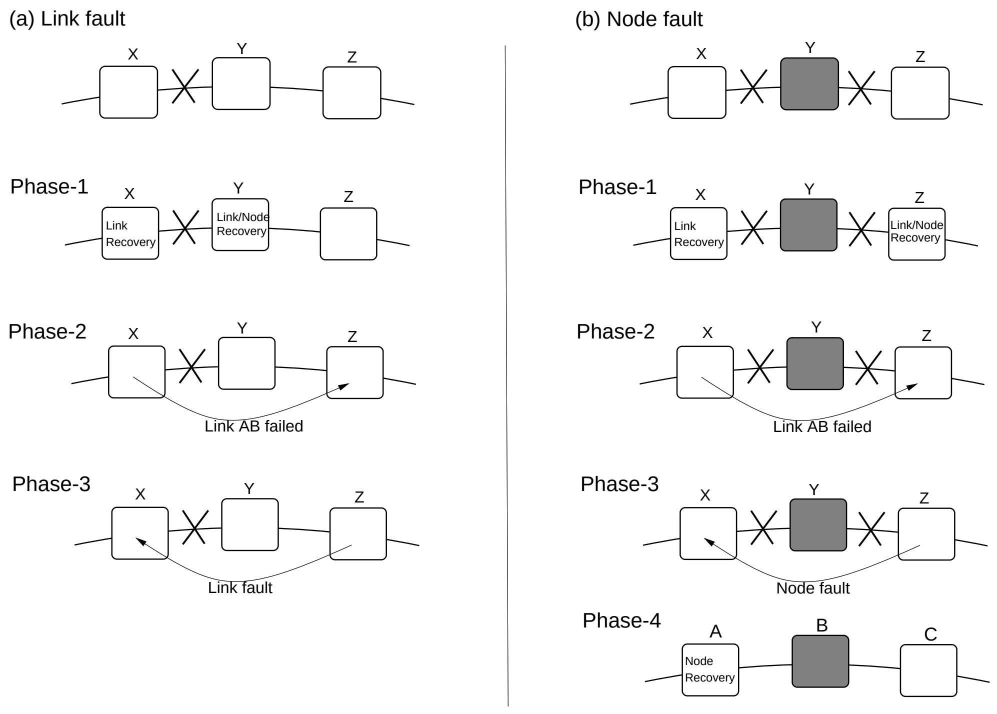


Figure 14: Integrated protocol for link and node faults

Phase 4. If X receives a link failure diagnosis, nothing needs to be done (X is already configured for this case). If X receives a node failure diagnosis it reconfigures to support this case.

This integrated method yields fast restoration times: in case of a channel fault, the diagnosis is based on local data and the recovery can be completed very quickly. In case of a link fault, recovery is performed as fast as the original scheme of Section 2.2; in case of a node fault, there is an added delay of a round trip from node X to Z.

4 Conclusions

In this paper we considered WDM ring networks with limited wavelength conversion capabilities in each node. While the paper mainly focuses on the O-E-O node architecture, most of it is applicable to the all-optical architectures depicted in Figure 2. The main part which should be modified for these architectures is the $W : 2$ protection mechanisms described in Section 2.1.

We considered three failure scenarios: single channel failures, link failures, and node failures. For channel failures we presented three solutions. One has small overheads but requires a major reconfiguration of a node in case of a fault affecting all the other channels. The second requires more hardware (an additional port per switch), and has longer paths to a backup channel ($\log_2 \frac{W}{2}$ instead of two hops), but requires only the faulty channel to reconfigure. Both solutions provide $W : 2$ protection. The third solution is integrated with link recovery and does not require additional hardware (over that of link protection) yet provides $1 : 1$ channel protection.

For link failures we proposed a solution requiring the faulty lightpaths to loop-back around the ring. This solution also supports multiple channel faults around the ring, provided that it is managed properly, and we presented a management scheme to ensure proper operation in case of multiple faults. The hardware overhead of the solution is low (one port per switch), but it requires 50% of the channels to be reserved for backup purposes (however, this is the minimum for any scheme), and long loop-back paths.

For node failures we presented two solutions. One is a simple extension of the configuration that handles link faults, requiring more hardware (25% more switches per node), and the other is based on sophisticated connection patterns between the nodes, requiring no additional hardware. We believe that the simpler solution is better suited to networks which require fast restoration, as it entails no coordination among the nodes.

Finally, we presented a scheme that combines all the recovery schemes, which we believe to be a practical integrated solution for such networks.

Acknowledgment. We would like to thank W. Eric Hall for many useful discussions.

References

[1] R. Ramaswami and G. Sasaki, "Multiwavelength optical networks with limited wavelength conversion," in *Submitted to IEEE Infocom'97*, 1997.

[2] F. Janniello, R. Neuner, R. Ramaswami, and P. Green, "Multi-protocol optical fiber multiplexer for remote computer interconnection," in *OFC'95 Tech. Digest*, 1995.

[3] T. Wu, *Fiber Network Service Survivability.* Artech House, 1992.

[4] T. Wu and R. Lau, "A class of self-healing ring architectures for SONET network applications," *IEEE Transactions on Communications*, vol. 40, pp. 1746–1756, Nov. 1992.

[5] I. Haque, W. Kremer, and K. Raychauduri, "Self-healing rings in a synchronous environment," in *SONET/SDH: a sourcebook of synchronous networking* (C. Siller and M. Shafi, eds.), pp. 131–139, New York: IEEE Press, 1996.

[6] Y. Hamazumi, N. Nagatsu, and K.-I. Sato, "Number of wavelengths required for optical networks with failure restoration," in *OFC'94 Tech. Digest*, pp. 67–68, 1994.

[7] N. Nagatsu, S. Okamoto, and K.-I. Sato, "Optical path cross-connect system scale evaluation using path accommodation design for restricted wavelength division multiplexing," *IEEE JSAC/JLT Special Issue on Optical Networks*, vol. 14, pp. 893–902, June 1996.

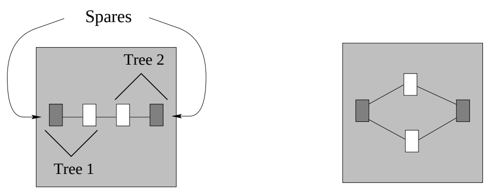
[8] R. Ramaswami and A. Segall, "Distributed network control for optical networks," *ACM/IEEE Transactions on networking*, 1996. Submitted.

[9] K.-C. Lee and V. Li, "Routing and switching in a wavelength convertible lightwave network," in *IEEE Infocom'93*, pp. 578–585, 1993.

[10] K.-C. Lee and V. Li, "A wavelength-convertible optical network," *IEEE/OSA Journal on Lightwave Technology*, vol. 11, pp. 962–970, May/June 1993.

[11] J. Yates, J. Lacey, D. Everitt, and M. Summerfield, "Limited-range wavelength translation in all-optical networks," in *IEEE Infocom'96*, pp. 954–961, 1996.

[12] S. Subramaniam, M. Azizoglu, and A. K. Somani, "Connectivity and sparse wavelength conversion in wavelength-routing networks," in *IEEE Infocom'96*, pp. 148–155, 1996.



(a) Result of simple scheme for W=4 (b) Result of optimal scheme

Figure 15: Comparison between configuration of the simple scheme and the detailed scheme

[13] A. Tucker, "Coloring a family of circular arcs," *SIAM Journal on Applied Math*, vol. 29, no. 3, pp. 493–502, 1975.

[14] V. Beneš, *Mathematical Theory of Connecting Networks.* Academic Press, 1965.

[15] D. Lawrie, "Access and alignment of data in an array processor," *IEEE Transactions on Computers*, vol. C-25, pp. 1145–1155, 1976.

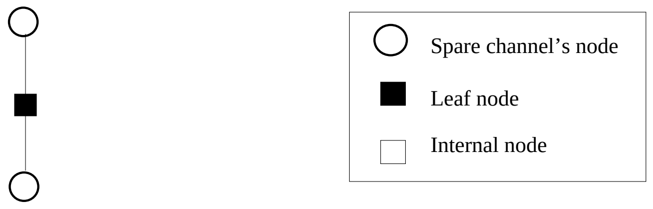
[16] D. Parker, "Notes on shuffle/exchange-type switching networks," *IEEE Transactions on Computers*, vol. C-29, pp. 213–222, Mar. 1980.

A Detailed scheme for constructing a dual tree

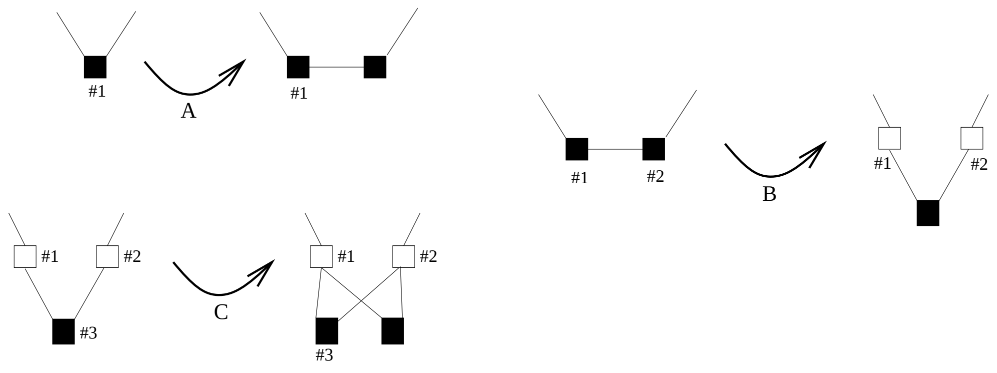
In this appendix we describe a detailed scheme for constructing the dual tree of Section 2.1.2. While the simple scheme presented there produces a good configuration, it is not optimal and the number of hops can be typically reduced by one, to obtain an optimal solution. This fact is demonstrated in Figure 15, in which the result of the simple scheme of Section 2.1.2 (Figure 15(a)) is compared to an optimal solution, which will be produced by the scheme described herein (Figure 15(b)).

In contrast to the simple scheme, wherein the dual tree was composed of two identical disjoint trees connected at the leaf level, here the trees are not disjoint and some leaves may be be common to both of them. The algorithm to construct a dual tree for W nodes (for W channels) is based on a set of $W-3$ transformations, starting from a basic, minimal configuration of three nodes and adding one node at a time. A detailed description of this process follows.

Basic configuration. Composed of a single leaf node (depicted as a full box), connected to the two spares (depicted as circles), see Figure 16(a).



(a) Minimal configuration



(b) Transformations

Figure 16: Basic configuration and transformations

Transformations. The basic configuration is extended one node at a time, by applying one of the transformations in Figure 16(b). In the figure, a transformation is depicted by an arrow, the spare nodes are depicted as circles, leaves are depicted as full boxes, while internal nodes are depicted as empty boxes. The transformations have the following priorities: (1) **C**, (2) **A**, (3) **B**. Hence, transformations of type **C** are applied whenever possible, only if no such transformations are possible, then transformation **B** is applied, and only if no transformations **C**, **B** are applicable, is a **A** transformation applied.

The transformations are fully described by the table in Figure 17, in which "Pri" stands for the priority of the transformations, "Node" stands for the increase in the total number of nodes in the configuration resulting from the transformation, and "Dist" for the maximal increase in the maximum distance from a leaf to a spare node.

An example for the construction of a 14 node dual tree is given in Figure 18. The feasibility of the

Trans.	Ord	Precondition	Description	Node	Leaf	Dist
A	2	(1) A single leaf #1 is connected to two internal or spare nodes. (2) Any other leaf x satisfying precondition 1 must be at least as far from the furthest spare.	Add a new leaf node as in the figure.	+1	+1	+1
B	3	Two leaf nodes #1 and #2 are connected to each other.	Insert a new leaf node #3 between nodes #1 and #2 (mutual to both trees). Nodes #1 and #2 are now defined as non-leaves (internal).	+1	-1	+0
C	1	There exist two internal nodes, #1 in one tree in the dual tree and #2 in the other, that are interconnected by less than $\Delta - 1$ leaves (the number of nodes as #3 is less than $\Delta - 1$).	Connect a new leaf node to both #1 and #2.	+1	+1	+0

Figure 17: Transformations for constructing a dual tree

construction is made possible by the following lemma.

Lemma 2. *There are enough free ports at the leaves to accommodate all the bypass connections from internal nodes.*

Proof. Note that each leaf has $\Delta - 1$ free ports (since it always has two connections to other nodes). Thus, if $\Delta \geq 4$, transformations **A** and **C** only enlarge the number of free ports (since only one node is added, and $\Delta - 2 > 1$ free ports are added). The only case which may deem the solution infeasible is transformation **B**, which decreases the number of available bypass ports at the leaves, while increasing the number of total nodes. This is taken care of by the precedence order of the transformations, which ensures that transformation **B** only takes place when there are extra ports at the leaves, to support the bypass connections. □

Lemma 3. *The maximal distance between any node to any spare node does not exceed* $1 + \log_\Delta \frac{W}{2}$.

The proof follows by simple induction on the size of the dual tree.

B Proof of Lemma 1

For the sake of simplicity assume $\Delta = 4$. It can be easily verified that if $\Delta > 4$ the proof still holds (if fact it becomes even easier).

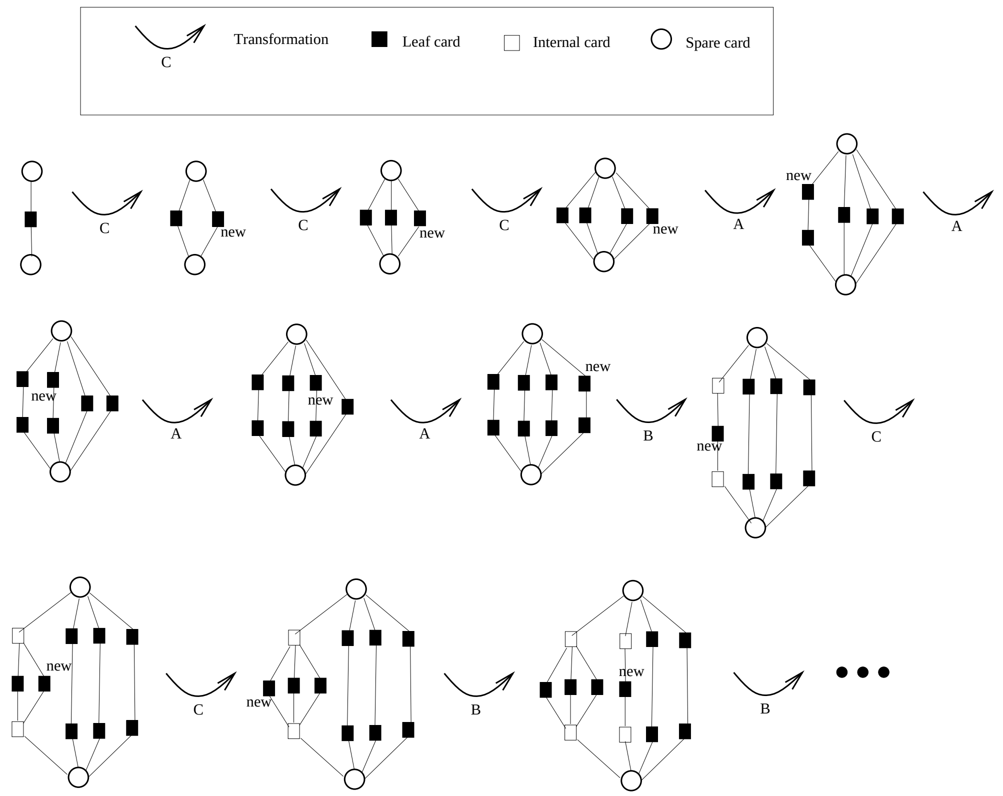


Figure 18: The construction of a 14 node configuration

In each of the two trees that comprise the dual tree, each node has at most two children (since they also have a connection to their parent in the tree and possibly a bypass link). As a result they are both binary trees. In fact, since these trees are populated as densely as possible, they are full binary trees, except maybe for the last level.

Let L_{last} denote the last full level of the tree, and let L_{higher} denote the set of nodes in all higher levels. Assuming the distance between L_{last} and the root is h, the number of nodes in L_{last} is 2^h. The total number of nodes in L_{higher} is $2^h - 1$. Note that only nodes in L_{higher} need bypass connections to leaves, as nodes in L_{last} are either leaves or are already directly connected to leaves. Now, each node in L_{last} can accommodate a bypass from a higher level node in the same tree, so all bypass links can use free ports of nodes in L_{last}.

If the trees that comprise the dual-tree were both full, this would have completed the proof. However, some of the nodes in L_{last} may not be leaves, and the bypass connections are not connected to them but to their children, which are leaves, shared by both trees. Note however that leaves have two free ports (two ports are occupied for connecting them to both trees), and can accommodate bypass connections from two L_{higher} nodes, one from each tree. $\square$