

Fig 3(a) Network consists of 4 nodes, 4 links and 3 lightpaths.

| Order | Demand |
|-------|--------|
| 1 | AD |
| 2 | CD |
| 3 | BC |
| 4 | AC |
| 5 | BD |

Table 3(b) Order in which lightpaths are routed

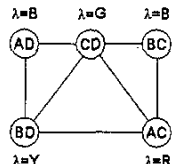


Fig 3(c) Online coloring for link D-C requires 4 colors (B,G,R,Y)

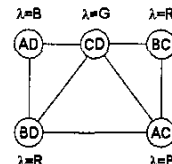


Fig 3(d) Optimized coloring for link D-C requires 3 colors (B,G,R)

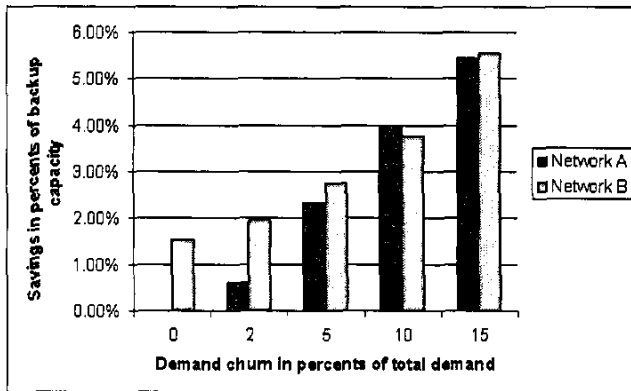


Figure 4.

paths. Since the order cannot be determined in advance, an optimization algorithm must be invoked at regular intervals to reassign the channels. In this write-up we show that finding the optimum assignment is equivalent to solving a vertex-coloring problem.

The allocation of protection channels is tantamount to a vertex-coloring problem: given the set of all restoration paths that intersect on a given link, represent every path as a vertex, and connect with an edge every pair of vertices whose corresponding paths are conflicting. Assign a distinctive color to each protection channel, and allot a protection channel to each path, that is color the vertices. Clearly, two vertices cannot be allotted the same color if they are connected by an edge, since the corresponding restoration paths are conflicting and cannot share a channel. The objective is to minimize the number of protection channels (respectively number of colors) required to accommodate all backup paths (respectively color all vertices), while avoiding conflicts.

This problem is known to be NP-hard, however there are many heuristics that can be used to compute sub-optimal solutions. A vertex-coloring algorithm that offers a good tradeoff between quality and runtime complexity is DSATUR[1].

Example

Consider the example of Figure 3 above (3a through 3d.) The figure illustrates five lightpaths {AD,CD,BC,AC,BD} and their protections, routed in a 4-node ring network. All the protections traverse link e_{CD} . The demands are provisioned following the sequence indicated in Table 3b. If we use a typical online shared mesh protection provisioning, and apply the graph representation presented earlier to e_{CD} , we obtain the "coloring" shown in Figure 3c. Even though a single failure in this example affects at most three primaries, this coloring consumes 4 colors, indicating that 4 protection channels are required. An optimized coloring yields the solution shown in Figure 3d, which consumes only 3 colors. Comparing 3c and 3d, we observe that a new channel (R) should have been allotted to the protection path of demand (BC) instead of sharing channel (B) with the protection of demand (AD). This solution however is not considered because not optimal when the third demand is being provisioned (that is {AD,CD,AC} are routed and {BD}

has not yet arrived) since at that time it would consume 3 channels instead of 2.

3. Implementation and Applications

The optimized channel reassignment is a low priority procedure. It can be a program thread running in background, or at regular intervals. The information necessary to accomplish this task is available locally in every OXC and independent of non-adjacent OXCs. Thus each OXC can run a copy of the algorithm in a distributed manner, locally and independently of other OXCs. A change in the allocation of a protection channel needs only to be propagated to its end-points. Since protection channels are "booked" and actually not cross-connected until a restoration occurs, the task amounts to no more than modifying and exchanging sharing databases between pairs of nodes. For every OXC-pair connected by at least one optical line, the OXC with highest IP address is delegated to perform the task.

A byproduct of the optimized channel reassignment is that it can be used to migrate the protection paths of mesh dedicated protections to shared mesh protections if desired. By changing their protection type to shared mesh protections, we allow the thread to apply the channel reassignment optimization to these services. The algorithm does not optimize the routes of the backup paths however, and the resulting solution is thus not as efficient as a re-optimization algorithm that re-routes the backup paths to maximize sharing[5].

4. Experiments

For our experiments we compare the benefits of local protection channel optimization on two realistic core mesh networks. Network A consists of shared-mesh capable optical switches in 46 cities interconnected by 75 fiber-trunks and loaded with 570 lightpaths. Network B consists of 61 switches, 88 fiber-trunks, and 419 lightpaths. For each network, we provision all the demands in sequence, using various values of demand churns (expressed in percent of the total demand), and perform a local channel optimization after all the demands are routed. We measure the amount of protection channel required before and after optimization and report the saving in % of total backup capacity in Figure 4. Our measurements indicate that as the demand churn increases, the

number of protection channels that can be freed becomes substantial.

5. Conclusion

This document proposes a distributed method that rearranges the allocation of shared channels reserved for restoration, with objective to minimize the number of allotted channels. This algorithm can be implemented as an independent background process to supplement existing provisioning algorithms. It is effective to correct sub-optimality inherent to a first fit based provisioning, or seize on improvement opportunities that are brought forth by demand churn.

6. References

- [1] Daniel Brélaz, "New Methods to Color the Vertices of a Graph," Communications of the ACM, Vol 22, Num 4. April 1979.
- [2] A. Akyamac, S. Biswas, J.F. Labourdette, S. Chaudhuri, K. Bala, "Ring Speed Restoration and Optical Core Mesh Networks," NOC'02, Darmstadt Germany, Jun. 2002.
- [3] J.F. Labourdette, E. Bouillet, R. Ramamurthy, G. Ellinas, S. Chaudhuri, K. Bala, "Routing Strategies for Capacity Efficient and Fast Restorable Mesh Optical Networks," Photonic Network Communications, pp. 219-235, July-Dec 2002.
- [4] G. Ellinas, et al., "Routing and Restoration Architectures in Mesh Optical Networks," Optical Networks Magazine, Jan-Feb 2003.
- [5] E. Bouillet, P. Mishra, J.F. Labourdette, K. Perlove, S. French, "Lightpath Re-optimization in Mesh Optical Networks," NOC'02, Darmstadt Germany, Jun. 2002.
- [6] S. Datta, et al, "Efficient Channel Reservation for Backup Paths in Optical Mesh Networks," IEEE GLOBECOM 2001, San Antonio, TX, Nov. 2001.
- [7] J. Doucette, W.D. Grover, "Comparison of Mesh Protection and Restoration Schemes, and the Dependency on Graph Connectivity," 3rd International Workshop on the Design of Reliable Communication Networks, Budapest, Hungary, Oct. 2001.

FQ3

11:15 AM

Capacity Requirements for Network Recovery from Node Failure with Dynamic Path Restoration

G. Shen, W. Grover, *TR Labs and Department of Electrical and Computer Engineering, University of Alberta, Edmonton, AB, Canada, Email: gshen@trlabs.ca.*

Node failure is not as frequent as span failure but recent events have emphasized its importance in network planning. We study the effects on capacity design if full or partial recovery from node failures is provided using failure-specific path restoration.

1. Introduction

Most studies of restorable networking consider span failures as the primary class of failure scenario. It is, however, often noted that because of its end-to-end orientation, a path restoration mechanism has an inherent ability also to respond to node failures. The spare capacity that ensures 100% span restorability is not necessarily adequate to ensure any particular target level of recovery from a node failure, however. Pre-planned shared backup path protection (SBPP) [1] does inherently protect transiting flows against node loss if primary and backup paths are all node disjoint. But SBPP also generally requires more spare capacity than dynamic path restoration and, due to its fixed pre-planned nature, has an inherently lower availability against to dual failure scenarios. It is of interest, therefore, to consider how much extra spare capacity an adaptive path restorable network needs to support node recovery, beyond that needed for span restorability. Other studies [2, 3, 4] have considered node recovery issues but to our knowledge the specific questions we ask, and the particular mechanism [6] and capacity design model [5] we consider are novel

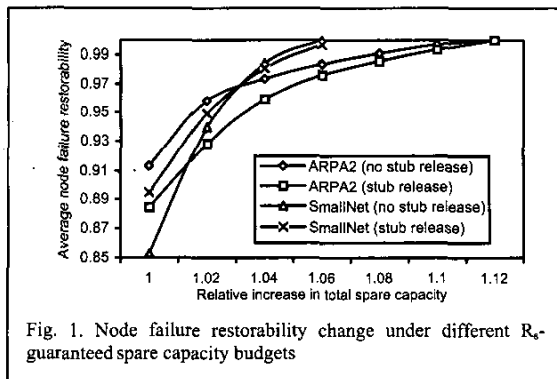


Fig. 1. Node failure restorability change under different R_s -guaranteed spare capacity budgets

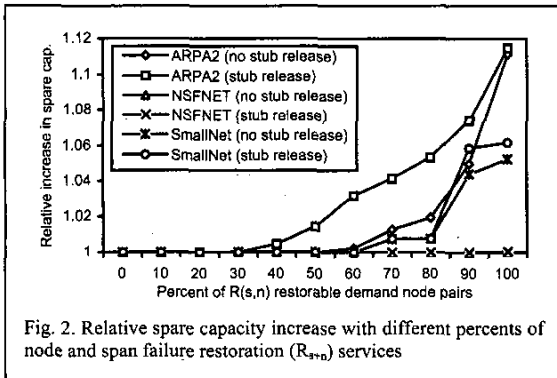


Fig. 2. Relative spare capacity increase with different percents of node and span failure restoration (R_{s+n}) services

and useful. Note that in studies on "node restoration," it is actually recovery of the transiting paths through a failed node that is intended. Service paths originating or terminating at the failed nodes cannot be restored by these methods. From one standpoint a node failure is like several concurrent span failures, suggesting large amounts of extra spare capacity would be needed. On the other hand, source/sink demands at the failed node are omitted from the recovery goal. So it is unclear in advance how much extra spare capacity is needed. With this in mind, the study poses three questions:

- Under suitable optimization, what are the maximum levels of node recovery that can be achieved with no more total spare capacity than required to ensure 100% span restoration?
- How much additional spare capacity is required to guarantee both 100% node and span restoration compared to span restorability only?
- How does capacity depend on the mix of services in a multiple Quality of Protection (multi-QoP) context (considering mixes of service with no protection, span failure protection, and both node and span protection)?

2. Methods and Results

To address these questions, we extend the prior model for dynamic adaptive path-restorable capacity design [5] and assume that there is at least enough wavelength conversion at each OXC node to make wavelength blocking insignificant. Three new design models were developed.

A) Design for 100% span and node failure restoration

This design model finds the minimal amount (and distribution) of spare capacity that guarantees 100% span restorability and 100% recovery of transiting demands at failed nodes. It is based on the conventional capacity design model for path restoration [5] with the addition of restorability and spare capacity constraints for each node failure scenario. This is the most straightforward (and potentially expensive) design approach.

B) Design to support Multi-QoP

This is an extension of the first model to consider three QoP levels. These are: (1) R_0 restorability: this is a wholly best-efforts class with no assured restorability, (2) R_s : this class is assured of restorability against any span failure, but receives only best efforts recovery for node failure. (3) R_{s+n} : A lightpath in this class enjoys assured restorability against any span or node failure (other than its own end-nodes). The design model places spare capacity so that all the affected traffic demands that require R_s or R_{s+n} restorability can be fully restored upon a span failure, and all the affected traffic demands that require R_{s+n} restorability can be fully restored upon a node failure. R_0 working capacity is effectively ignored in the spare capacity design problem, but would receive best-efforts restorability in real time within the spare capacity remaining after re-routing for R_s or R_{s+n} service classes.

C) Maximal node recovery under a spare capacity budget

This design model allows us to set a budget on total spare capacity (above that where 100% span restoration is feasible) and optimize the distribu-

tion of spare capacity for the highest achievable node failure recovery level as well. In Models A and B the objective function is minimum cost or capacity, but here it is to minimize the total *un-restored transiting demand over all node failures*. By varying the budget amount this model can be used to systematically study the trade-off between cost and node recovery level.

Each of these design models is also implemented with and without "stub release" (the conversion of surviving channels on failed paths into available spare capacity as per [5,6]). In the context of node recovery, stub release means that the surviving lightwave channels of paths transiting the failed node (which will be restored) and of paths sourced at the failed node (which will not be restored) are both available as spare capacity for the restoration effort. This is a failure-specific restoration response to exploit available capacity, in contrast to the fixed response of SBPP-planned paths to a failure anywhere along their route. The failure-specific re-routing may be either found adaptively in real time [6], or also pre-planned and stored in the OXC nodes.

We evaluate the performance of the design models on five well-known topologies: ARPA2 (21-nodes 25-spans), NSFNET (14,21), SmallNet (10,22) from [5], Cost 239 (11,26), and the (55, 63) topology from www.level3.com. Lightpath demands were generated following a uniform random distribution on the range [1...20] for each node-pair. For the larger Level3 test case only the largest 30% of demand pair volumes are employed for computational reasons. Note that this demand model makes demands between distant nodes as intense as between neighbors, creating relatively large numbers of transiting paths at nodes. This is the more challenging case for node recovery compared to demand models with a gravity-like localization effect. A single shortest route is used for each working demand but the design models consider all possible distinct routes between each node pair for each failure scenario for spare capacity placement. The design problems were all solved within half an hour with AMPL/CPLEX 7.1, except Level3, which took several hours.

Table 1 summarizes the results. The first (double) row is obtained using Model C to maximize node recovery given a budget equal to that of a span-restorable design only. This represents the intrinsic potential for node restorability with no more capacity than needed for conventional span failure restoration. Note how high this "free" node recovery capability can be - almost 100% in many cases and always at least 78%. The recovery levels are lower with stub release (at first somewhat counter-intuitively) because the basic networks to handle span failures only also have significantly less spare capacity when using stub release. The second and third rows show the relative amounts of extra spare capacity and total cost increase for the network to wholly support R_{s+n} services, as compared with the network supporting R_s services only. The values are surprisingly marginal. Consistent with the very high intrinsic R_n , NSFNET and Level3 require almost no added capacity to support 100% node recovery. And in no case was more than 10% extra spare capacity needed to ensure 100% node recovery. These findings suggest that the "abandonment" of source/sink paths at the failed node in measuring node recovery levels is a more dominant effect than the fact that node failure is like multiple span failures, especially on sparse topologies.

Fig. 1 shows the results of using Model C to study node recovery levels with various total spare capacity budgets. Starting from the point of the spare capacity designed just for 100% span restoration, a very small budget increase gives a great improvement of node failure restorability. For example, there is 9% restorability improvement when there is 2% spare capacity increase for the ARPA2 network without stub release. However, the improvement rate tends to slow down after that. For SmallNet without stub release, a budget increase from 4% to 12% can only bring about 3% restorability improvement. The explanation seems to be related to the approach to 100% R_n itself. Under a small budget, there is a large num-

Table 1. Node failure restorability, redundancy increase, and total cost increase for various network design cases

| Networks | | ARPA2 | NSFNET | SmallNet | Cost239 | Level3 |
|---|-----------------|--------|--------|----------|---------|---------|
| Intrinsic node failure restorability | No stub release | 91.35% | 99.84% | 85.30% | 78.89% | 95.59% |
| | Stub release | 88.43% | 99.60% | 89.40% | 82.85% | 99.998% |
| Redu. increase (R_{s+n} vs. R_s) | No stub release | 10.0% | 0.02% | 2.6% | 3.4% | 4.1% |
| | Stub release | 9.7% | 0.03% | 2.8% | 1.4% | 0.1% |
| Total cost inc. (R_{s+n} vs. R_s) | No stub release | 5.2% | 0.01% | 1.7% | 2.4% | 2.0% |
| | Stub release | 5.3% | 0.02% | 1.9% | 1.0% | 0.001% |

ber of un-restorable lightpaths, hence many chances to make improvements. At higher budgets there are fewer un-restored lightpaths remaining to be made restorable in any "easy" way. Finally Fig. 2 shows the results on the relative spare capacities required to support multi-QoP services. For simplicity of presenting results we assume a mix of only R_s and R_{s+1} services (no R_0). We find that the spare capacity designed for R_s restorability alone can serve up to 30% (for ARPA2 with stub release), 50% (for ARPA2 without stub release), 60% (for SmallNet), and almost 100% (for NSFNET) R_{s+1} services in the networks as well. This implies that without any additional spare capacity over the conventional designs, a network can still serve a large number of higher-level QoP services.

3. Concluding Remarks

This work shows that, overall, it is surprisingly easy to support node recovery in path-restorable networks. Very high levels of premium service class guarantees (services assured of both span and node recovery) can apparently be supported with no more spare capacity than needed to give all services span restorability alone. Conversely if 100% node recovery is desired by design it took at most 10% extra spare capacity to provide this. This knowledge and related design methods are useful in themselves and to further inform the comparison of failure-specific path restoration to the SBPP pre-planning scheme.

4. References

- [1] W.D. Grover, J. Doucette, "Design of a meta-mesh of chain sub-networks: ...," *JSAC* 20, 47-61 (2002).
- [2] A. Chiu, J. Strand, "Joint IP/optical layer restoration after a router failure," *OFC 2001*, pp. MN5/1 -MN5/2.
- [3] D. Stamatelakis, W.D. Grover, "IP layer restoration and network planning based on virtual protection cycles," *JSAC* 18, 1938-1949 (2000).
- [4] Y. Liu, D. Tipper, "Successive survivable routing for node failures," *Globecom 2001*, pp. 2093-2097.
- [5] R.R. Iraschko, M. MacGregor, W.D. Grover, "Optimal capacity placement for path restoration in STM or ATM mesh survivable networks," *IEEE Tran. Net. 6*, (1998), 325-336.
- [6] R.R. Iraschko, W.D. Grover, "A highly efficient path-restoration protocol ...," *JSAC*, vol.18, no.5, May 2000, pp. 779- 793.

FQ4

11:30 AM

FASTeR: Shared Restoration Without Signaling

V. Poosala, C. Phadke, *Bell Labs, Lucent Technologies, Murray Hill, NJ*; A. Shandilya, *Univ. of Rochester, Rochester, NY*, Email: poosala@research.bell-labs.com.

We present a novel technique for rapid restoration in optical/electrical mesh networks that does not incur signaling or cross-connect setup latencies. We show that it easily meets the 50ms requirement in most scenarios.

Introduction

The last few years have witnessed the introduction of optical and electrical mesh networks as an alternative to SONET ring networks. One of the key benefits of mesh networks is the improved bandwidth utilization coming from shared restoration. Unlike the traditional 1+1 protection schemes which reserve 50% of the bandwidth for protection, shared restoration allows multiple demands to share backup links and hence reserves less capacity. However, the speed of restoration is an issue with this scheme. Many classes of traffic, especially voice traffic, require paths to be restored very fast, often within 50ms [2]. It is easy to meet this requirement with 1+1 protection - data is always sent on the primary and backup paths and the end node picks the best signal. This is not the case with shared restoration where the backup paths are only set up after the failure. Typ-

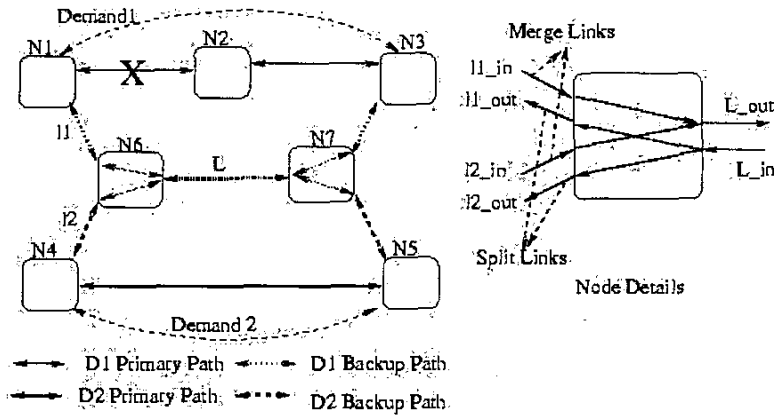


Figure 1: Split Merge Node Configuration.

ically, this process involves signaling to setup cross-connects(XCs) which can be time-consuming in large networks using slower XC technologies.

Motivated by these issues, we have developed a rapid restoration mechanism that works for both optical and electrical mesh networks, with a primary goal of eliminating the signaling and XC latencies.

Related Work: Much of the earlier work on shared restoration has focused on routing and design algorithms and our technique is complementary to those results. A mechanism called ROLEX for fast restoration was proposed in [1] and its implementation in GMPLS/RSVP was given in [3]. However, ROLEX also incurs cross-connect and signaling latencies, which are the primary bottlenecks addressed in our solution.

Solution

We consider a mesh network consisting of cross-connect nodes (XCs) connected by DWDM systems. Each wavelength between two adjacent nodes is considered to be a link. The edge nodes are assumed to have OEO capability. Traffic consists of unit-wavelength demands between the edge-nodes, protected through shared restoration. Each demand is associated with a *primary path* and a precomputed *backup path* for the entire path or per link. For clarity, we assume bi-directional traffic using the same path in both directions and bi-directional failures for our discussion and present the modifications for uni-directional cases where necessary.

The basic idea behind our solution, called FASTeR (FAST Signal-free Traffic Restoration), is as follows. Even before any failure occurs, all the backup paths are set up using certain special features of the XCs, in contrast with the traditional approach where the backup paths are only pre-computed but not actually setup until failure occurs. On failure, the end-nodes of the failed demand immediately start sending data on the backup path, without any prior signaling. The details are described next.

We present our solution to cover a wide-range of generic hardware with the following characteristics, rather than assuming specific XC technology like 3D MEMS.

1. *Coupling/Multicasting:* The node can merge multiple in-links (*merge links*) onto the same out-link; and multicast data from an in-link to multiple out-links (*split links*).

2. *Selection:* The node will send light from only one of the merge links onto the out-link. One of the ways this may operate is as follows. All the merge links are in a "non-blocked" state when there is no light on any of them. When there is light on one of them it goes onto the out-link and in parallel the node blocks the others within certain *blocking time*. The links remain blocked until the light stops. There is a chance for garbling the data if light comes on the other links before they

are blocked, which is handled by our scheme.

Blocking is known to be a simpler operation than setting up a XC and is expected to take much less time. E.g., it is possible to achieve blocking times in nanoseconds using LiNbO3 and in micro-seconds to few milliseconds using ferro-electric crystals, MEMS [5]; however, our scheme can tolerate much higher blocking times as shown later.

Note that many of the current XCs already provide a similar "bridge-merge" functionality to support 1+1 protection, typically combining up to two or four ports. The main focus of this paper, however, is on the overall scheme; more specific details of the hardware are outside its scope.

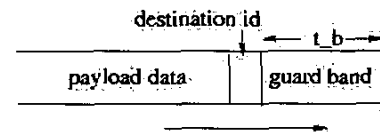


Figure 2: Message Format.

Path Setup: When a demand arrives, its backup path can be computed using any routing algorithm and the XCs are configured as follows. Consider a node where two or more backup links $l_1 - l_n$ merge onto a single backup link L . The links $l_1 - l_n$ in the incoming fiber are configured as merge links, merging onto the outgoing L . The links $l_1 - l_n$ in the outgoing fiber are configured as split links, with light from the incoming L multi-cast onto them. This is illustrated in Figure 1. By default, the backup links carry no light and hence all the merge-links are non-blocked. A protocol like RSVP can be extended to implement the necessary messages.

Restoration: On a network failure, the end-nodes of the failed demand (N1, N3 in the figure) send data out on their backup paths, preceded by a *guard band* equal to the blocking time and a header containing the destination node identifier (see Figure 2). When this light reaches an intermediate node over a merge link, it will be sent over the corresponding shared link while blocking the other merge links. If there are near-simultaneous failures in the network, multiple transmissions may arrive on the merge links before blocking is completed. This can result in garbling of data for that time duration. By choosing a guard band equal to the blocking time, the corruption is restricted to the guard band. The signal is multi-cast onto the split links when encountered, ultimately reaching the other end-node. On receiving light on the backup link, the end node skips the guard band, reads the header, and picks up the data only if the id in the header matches its own.

Essentially, blocking replaces the more complex cross-connect setups and multi-casting eliminates the need to process address headers in the internal nodes.

Note that, due to the multi-casting, data may go on parts of the network that were not in the backup