A Partial-Protection Approach Using Multipath Provisioning

Ananya Das, Charles Martel, Biswanath Mukherjee
Department of Computer Science, University of California, Davis, CA 95616
Email: {das, martel, mukherje}@cs.ucdavis.edu

***Abstract*—We study the problem of reliably provisioning traffic using multipath routing in a mesh network. Traditional approaches handled reliability requirements using full-protection schemes. Although full-protection approaches offer high assurance, this assurance can be costly. We take a less expensive approach to maintain reliability by offering *partial-protection*. Specifically, our approach guarantees part of the requested bandwidth, rather than the full amount, in the event of a link failure. We first show that the amount of partial-protection that can be guaranteed is limited by the topology of the network and the bandwidth requirement of a connection request. We then propose an effective multipath algorithm that attempts to provision bandwidth requests while guaranteeing the maximum partial-protection possible. Results show that by effectively selecting paths that limit edge overuse, our algorithm achieves very low bandwidth blocking probability. Our algorithm also serves significantly more requested bandwidth than the protection approach.**

I. Introduction

Network service providers typically must provision connection requests while satisfying Quality-of-Service (QoS) requirements. An important QoS metric is service reliability. Reliability may be measured by how efficiently an application can deal with network failures. Traditional approaches deal with failures using *full-protection* mechanisms such as path *protection* [2, 5, 8, 12, 13, 17] and path *restoration* [3, 6, 10, 11, 13]. Although these approaches offer full recovery in case of network failures, this assurance can be expensive.

Instead, network providers can deal with failures by providing part of the requested bandwidth, rather than the full amount, in case a network failure occurs. This approach is referred to as *partial-protection*. Since partial-protection uses fewer network resources, network providers may offer it to customers who do not require full-protection, and some customers will find partial-protection an attractive option if offered at a lower cost.

The term *partial-protection* has been used to refer to various reliability-aware provisioning schemes [4, 15, 16]. Some of these studies have considered link-level protection [15, 16], however, in this paper we study path-level partial-protection [4]. Specifically, we define *partial-protection* as a bandwidth provisioning approach that guarantees a specific percentage of the requested bandwidth if a link failure occurs.

In the online partial-protection provisioning problem we study, bandwidth requests arrive dynamically and must be scheduled as they arrive. Each request is associated with a partial-protection requirement, which is the fraction of bandwidth that must be guaranteed, even in case of failure. For each request, we must provision the requested bandwidth while satisfying the partial-protection guarantee. We consider this problem for high-capacity backbone mesh networks supporting virtual concatenation (VCAT) [18]. VCAT enables a connection to be inversely multiplexed on to multiple paths, a feature that has many advantages over conventional single-path provisioning. Multipath provisioning provides better fault tolerance, more effective utilization of network resources, and reduces link congestion.

We show that the level of partial-protection possible is limited by the network topology and the bandwidth request. To our knowledge, this is the first work to identify these limits. We then propose an online multipath heuristic to solve this problem. Our algorithm tries to satisfy the maximum amount of partial-protection that can be guaranteed. Unlike previous studies [4, 14], our algorithms guarantee a *specific* level of partial-protection and can be used to provision connection requests regardless of the network topology and bandwidth requirement. Our algorithm yields low bandwidth blocking probability even at heavy load.

In this paper, we study a setting where customers are willing to accept partial-protection, rather than full-protection, in the (rare) case of a network failure. Although full-protection approaches provide complete service even in case of failure, they often require significant network resources. This may result in a decline in overall performance, particularly an increase in bandwidth blocking probability. On the other hand, our algorithm satisfies connection requests while using fewer resources than protection schemes. To understand the cost of full-protection on performance, we compared our algorithms to an efficient full-protection multipath algorithm proposed in [12]. (we describe this heuristic in more detail in Section II). Our results show that our algorithm can provision more of the requested bandwidth than the full-protection scheme. We also find that using a mixture of full and partial protection is more effective than using only full-protection. These results verify that there exists a significant trade-off between full-protection and performance. Therefore, if some customers are willing to accept partial-protection in the (rare) event of a failure, then our approach would be beneficial.

The remainder of this paper is organized as follows. Section II describes some other reliability-aware approaches. Section III describes the provisioning problem that we study and our proposed solutions. Section IV provides our evaluation results. Finally, Section V presents our conclusions.

This work has been supported by NSF Grant No. CNS-0832176.

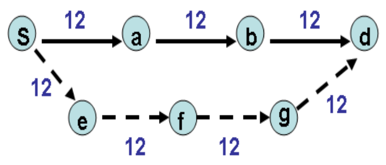

Fig. 1. Example of full path protection. Solid lines indicate the primary path and dashed lines indicate the backup path. To satisfy a request of 12 units of bandwidth from s to d, 84 units are consumed from the network.

II. Related Work

Path *protection* [2, 5, 8, 12, 13, 17] and path *restoration* [3, 6, 10, 11, 13] are two common approaches used to handle network failures. (For a thorough review of these techniques, please refer to [7].) Path protection is a proactive procedure in which spare capacity is reserved during connection setup so that when a link on the primary path fails, the connection is rerouted over a link-disjoint backup path (see Fig. 1). In [12], the authors propose an effective protection heuristic which achieves fast fault-recovery time and moderate backup sharing by judiciously allowing primary paths to share backup capacity. Although this heuristic is an efficient protection approach, in Section IV, we show that maintaining full-protection may cause an increase in bandwidth blocking probability.

Path restoration is a reactive procedure in which backup paths are discovered after a failure occurs on a primary path. Restoration schemes take more time to restore a connection than protection schemes since recovery is performed after the failure occurs. Therefore, for time-critical applications, protection schemes are often a more attractive approach.

The authors of [4] also use partial-protection to deal with link failures. In their setting, the partial-protection guarantee is specified by the user. If they cannot meet the guarantee due to the current network state, then they provide the maximum partial-protection possible. One drawback to this approach is that the level of protection that is ultimately provided to the user is limited by the efficiency of the routing scheme. For example, if a good routing scheme is used, then more requests will be satisfied initially, and more network resources will be consumed. Then it will be more difficult to provide high levels of partial-protection for future requests.

The authors of [14] also use a partial-protection algorithm to deal with network failures (they refer to their approach as *degraded-service*). However, they make stringent assumptions about the types of requests that their algorithm accepts.

III. Provisioning with Partial-Protection

In the *partial-protection provisioning* (PPP) problem that we study, bandwidth requests arrive dynamically and each request must be scheduled (or rejected) as it arrives. Each connection request consists of a source, destination, bandwidth requirement, and *partial-protection guarantee*, which is the fraction of bandwidth that must be guaranteed even if a link failure occurs (in Sec. III-A we discuss possible partial-protection values). The goal of the PPP problem is to find a set of paths from the source to the destination that satisfies the bandwidth and partial-protection requirements.

The PPP problem takes as input a directed graph $G = (V, E)$, where V is a set of vertices and E is a set of edges such that each edge in E has a non-negative integer capacity. Connection requests arrive dynamically, and each request is of the form $< s, d, b, f >$, where $s, d \in V$, s is the source node, d is the destination node, $b \in B$ is the bandwidth requirement, and $f \in (0, 1)$ is the partial-protection guarantee. The goal of the PPP problem is to find a set of paths from s to d such that (1) the bandwidth from s to d is at least b, and (2) if a link failure occurs, at least $b \cdot f$ is still available on the surviving paths.

Figure 2(a) shows an example of 50% partial-protection. For this example, suppose the request $< s, d, 12, 0.5 >$ has been issued. Since 50% partial-protection must be guaranteed, then with 2 link-disjoint paths, every link can have at most flow 6. Using partial-protection to provision this request consumes 42 units[1] of bandwidth from the network, whereas using protection would consume 84 units (see Fig. 1).

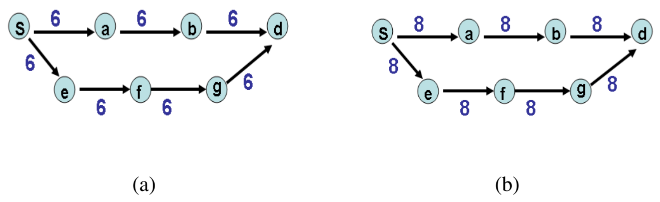

Fig. 2.(a)Example of 50% Partial-Protection. To satisfy request $< s, d, 12, 0.5 >$, 42 units of network bandwidth are consumed. (b) With Over-Provisioning we can satisfy 66% partial-protection.

A. Determining Appropriate Partial-Protection Values

For a given source, destination, and bandwidth requirement, the maximum partial-protection that a service provider can guarantee is limited by the network topology and the bandwidth requested. Let $< s, d, b, f >$ denote a connection request. To satisfy the bandwidth requirement, we must find a set of paths from s to d with a flow of at least b. To satisfy the partial-protection guarantee, we must ensure that f is set such that no link carries more than $b(1-f)$ flow. Let m denote the size of the minimum edge cut between s and d in the original graph G. Therefore, we know there are at most m disjoint paths between s and d in the original graph, though fewer may exist later as edges are used. Thus, to send b units, some path must carry at least $\lceil \frac{b}{m} \rceil$, so at least this amount will be lost if an edge fails. To satisfy the partial-protection guarantee, we must ensure that at most $b(1-f)$ units of flow are lost if a link failure occurs. Therefore (with unit granularity for bandwidth requests and link capacities), we have:

$$\left\lceil \frac{b}{m} \right\rceil \quad \leq \quad b(1-f) \tag{1}$$

Therefore,

$$f \quad \leq \quad 1 - \frac{\lceil \frac{b}{m} \rceil}{b} \tag{2}$$

In other words, Eqn. 1 states that the fraction of bandwidth lost is at least the minimum amount of flow that may be

[1]One unit of bandwidth in a SONET-based network is STS-1 ($\approx$ 51.84 Mbps).

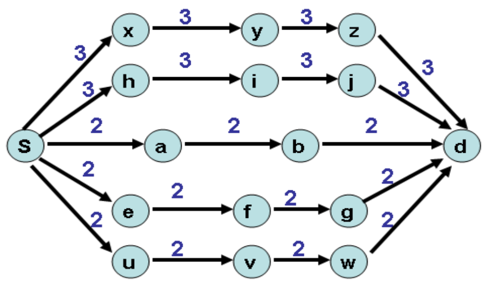

Fig. 3. Example of 75% of Partial-Protection. In this example, $b = 12$, $m = 5$, so $f = .75$.

allotted to a single path (In Sec. III-D, we describe a technique which allows us to satisfy higher partial-protection by allocating more bandwidth).

Figure 3 shows an example of how the network topology and the bandwidth request limit the maximum partial-protection that can be guaranteed. For this example, assume that a request for 12 units from s to d arrives. Since the size of the minimum $s-d$ cut is 5, some path must carry at least $\lceil \frac{12}{5} \rceil = 3$ units. Therefore, at least $\frac{3}{12}$ of the flow could be lost if a link failure occurs, so at most 75% service can be guaranteed. We can provision this request by allotting a flow of 3 on two paths and a flow of 2 on three paths. Note that additional capacity and integrality constraints can further restrict the value of f. For example, if paths *s-a-b-d* and *s-e-f-g-d* each had free capacity of 1, then at least one other path would need to carry a flow of at least 4, which would restrict f to $1 - \frac{4}{12} = \frac{2}{3}$.

B. The MDP Algorithm

We propose the MIN-CUT DISJOINT PATHS (MDP) algorithm to solve the partial-protection provisioning problem. For a pair of nodes s, d, let m denote the size of the minimum $s-d$ cut in the original graph. Let b denote the bandwidth requirement and let f denote the maximum partial-protection that can be guaranteed. Recall that f is the percentage of b that should survive in case a single link failure occurs. As shown in Eqn. 2, f is constrained by both m and b. When request $< s, d, b, f >$ arrives, MDP looks for a set of m paths from s to d such that the total flow along the paths is at least b while maintaining that no link carries more than $b(1-f)$ (In Sec. III-D we describe a technique to use if m such paths are not available). The latter constraint ensures that $b \cdot f$ units of flow survive if a link fails. MDP looks for a set of disjoint paths since this ensures that no single link carries more flow than it should. To find a good set of disjoint paths, MDP uses the path-searching technique proposed in [9]. The basic idea is to first find a set containing all shortest disjoint paths between the source and destination. Then by efficiently splicing together sub-paths, we can create a large collection of sets of short disjoint paths. MDP uses this technique to create a collection of sets of disjoint paths that satisfy the bandwidth and partial-protection requirements. It then chooses the "best" set to route the connection. We consider the best set to be the one that consumes the least amount of network bandwidth. If the algorithm cannot find a set of disjoint paths with enough free capacity, it rejects the request.

C. Smarter MDP Algorithm

Congestion is common when routing a series of network requests. Edges that lie on the shortest paths for many node pairs are likely to be used frequently. To avoid the overuse of these "popular" edges, we modified the MDP algorithm so that it preserves bandwidth on frequently used edges. In the improved algorithm, SMART-MDP, we initially assign the edges in the graph a unit cost. We then use the path-finding techniques from [9] to find a set of cheap disjoint paths between the requested source and destination. Each time an edge is selected, we increase its cost. Therefore, popular edges will have higher costs and will be used less frequently for future requests. These edges will be used mainly when they are crucial for efficiently satisfying a request. Similarly, when a request departs, we decrease the costs of the edges used to route the request. Adjusting the costs of edges allows us to easily tune the algorithm based on the network topology and frequency of edge accesses. This feature of SMART-MDP makes it a flexible approach since we can control the congestion of any particular edge. It also allows our algorithm to adapt to changing network conditions and user traffic patterns. We describe SMART-MDP in more detail in Algorithm 1.

Algorithm 1. SMART-MDP($G = (V, E), C : E \rightarrow Z^+$)

```
1:  Assign each edge $e \in E$ a unit cost.
2:  **for all** requests $< s, d, b, f >$: **do**
3:      Use the approach from [9] to find a large set, $P$ of
        cheap disjoint paths from $s$ to $d$.
4:      Find a set of paths $p^* \in P$ such that:
5:      (1)the total flow on the paths in $p^*$ is at least $b$
6:      (2)no link carries more than $b(1-f)$
7:      (3)the total amount of bandwidth consumed by $p^*$
        is minimized over all sets in $P$ satisfying (1) and (2).
8:      **if** such a set exists **then**
9:          Increment the cost of all edges in $p^*$.
10:         Reduce the capacity of every link in $p^*$ by its new
            flow.
            Provisioning Successful.
11:     **else**
12:         Reject this request.
13: When request $< s, d, b, f >$ departs:
14: Decrement the cost of all edges in $p^*$ and restore
    capacity on these edges.
```

D. Partial-Protection with Over-provisioning

Given a request, if we cannot find enough disjoint paths to meet the bandwidth and partial-protection requirements, we can still satisfy the request by *over-provisioning*. In other words, by provisioning more than the bandwidth request, we can satisfy higher partial-protection guarantees. For example, in Fig. 2(b), we could satisfy the request $< s, d, 12, 0.66 >$ by allocating 8 units each on paths *s-a-b-d* and *s-e-f-g-d*. Although over-provisioning may require allocating more than the requested bandwidth, we found that this technique yields notable performance improvements (see Fig. 7). Therefore, we use over-provisioning in our algorithms.

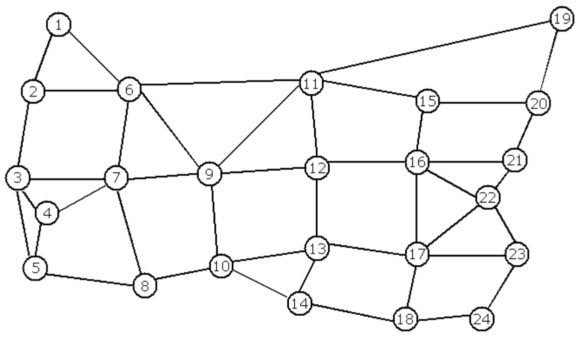

Fig. 4. Sample network topology.

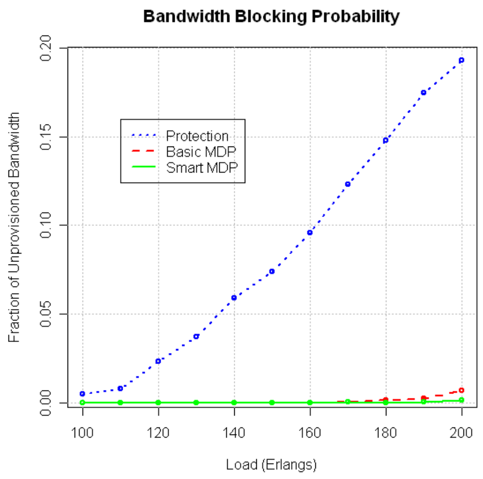

Fig. 5. 50% Partial-Protection: Protection vs. MDP vs. SMART-MDP.

IV. Illustrative Numerical Results

To evaluate the performance of our algorithms, we simulated a realistic dynamic network environment used in previous studies [9, 12]. The connection arrival process is Poisson and the connection-holding time follows a negative exponential distribution with unit mean. The network is fully wavelength convertible. There are 16 wavelengths per link, and the capacity of each is OC-192 ($\approx$ 10 Gbps), a realistic measure for today's channel speeds. The bandwidth occupies an integral number of slots with STS-1 granularity($\approx$ 50Mbps). The bandwidth is distributed as follows: 51.5% of the requests are for 50 Mbps of bandwidth, 25% are for 100 Mbps, 10% are for 150 Mbps, 5% are for 600 Mbps, 5% are for 1 Gbps, 2% are for 2.5 Gbps, 1% are for 5 Gbps, and the final 0.5% of requests are for 10 Gbps of bandwidth. We simulated 100,000 connection requests for various load levels[2]. We used the sample topology shown in Fig. 4.

A. Fixed Partial-Protection Guarantee

Network providers may offer a *fixed* amount of partial-protection to all customers who do not have stringent protection requirements. Therefore, we first study a network setting where the partial-protection guarantee is fixed at 50% for all connections. With unit granularity, it is not possible to satisfy requests for 1 unit with partial-protection. Similarly, requests for 3 units cannot be satisfied with 50% partial-protection if there are only 2 edge-disjoint paths between the source and destination. To satisfy the 50% guarantee, we would have to maintain that all paths carry at most a flow of $\lfloor 0.5{*}3 \rfloor = 1$ unit. With this constraint it is not possible to achieve a flow of 3. When requests cannot be satisfied with 50% partial-protection, our algorithms provide additional protection. Specifically, for 1 unit requests, we look for 2 disjoint paths each carrying 1 unit of flow. Similarly, for the 3 unit requests, we look for 2 disjoint paths each carrying 2 units of flow. For all other requests, we look for a set of disjoint paths and select the best set. If we cannot find a set of paths with enough free capacity, we use over-provisioning to route the request.

We tested MDP and SMART-MDP and compared the results to an efficient protection algorithm[3] [12] which we described in Sec. II. The results, in Fig. 5, show that our partial-protection algorithms block significantly less bandwidth than the full-protection approach. Even under a high load level of 200 Erlangs, our algorithms satisfy almost 20% more bandwidth than the protection approach and reduce the bandwidth blocking probability by more than 96%. Under this load, MDP can satisfy 99.3% of the requested bandwidth while SMART-MDP can satisfy 99.9%. Note that unlike the competing protection scheme, for our algorithms, there is no additional overhead caused from backup sharing. However, if our algorithms allowed backup sharing, as the competing protection approach does, they would have even lower blocking probabilities.

B. Maximum Partial-Protection

We now consider a network setting where the network operator must provide the customer with the maximum amount of partial-protection that can be guaranteed. Since many customers would be unsatisfied with less than 50% protection, we require the protection guarantee be at least 50%. As in the previous case, our algorithms provide full-protection for requests that cannot be satisfied with at least 50% partial-protection. Based on Eqn. 2, for our topology and bandwidth distribution, we provided full-protection for 51% of the requests, .75 to .79 partial-protection for 3% of the requests, $\frac{2}{3}$ partial-protection for 18% of the requests, and .5 partial-protection for 28% of the requests. As in the previous simulations, we use over-provisioning when necessary. Figure 6 shows the results of our algorithms compared to the protection scheme. Since under this setting, we must provide higher levels of partial-protection than in the previous setting, MDP and SMART-MDP yield higher bandwidth blocking probabilities. However, our algorithms still perform significantly better than the protection approach. Under high load (200 Erlangs), MDP reduces the bandwidth blocking probability by 72% while SMART-MDP reduces it by 87%. Figure 6 also shows that by selecting good paths

[2]Load, measured in Erlangs, is defined as the product of the connection arrival rate, the average connection-holding time, and a connection's average bandwidth normalized to units of OC-192.

[3]In[12], two shared protection heuristics are proposed: the PREV algorithm achieves fast fault-recovery time; the PIVM algorithm services more requests, but requires much longer fault-recovery time. Since our algorithms do not require any path switching, they achieve fast fault-recovery time. Therefore, to present a fair comparison, we compare our algorithms to the PREV heuristic which also achieves fast-fault recovery.

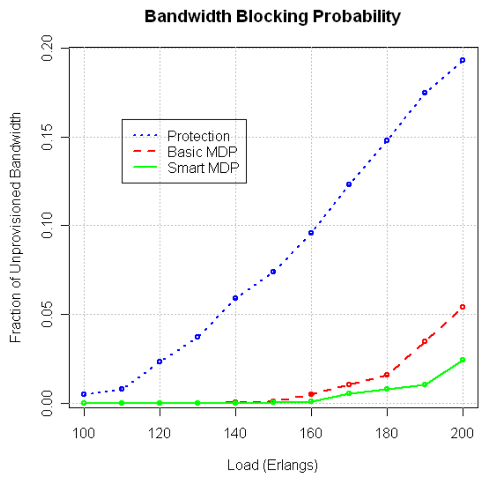

Fig. 6. Maximum Partial-Protection: Protection vs. MDP vs. SMART-MDP.

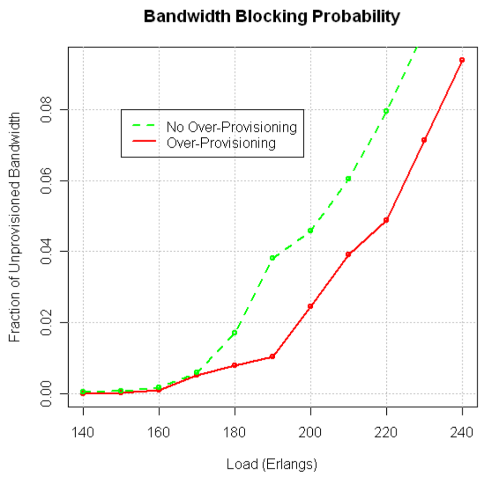

Fig. 7. Performance Gain Achieved with Over-Provisioning (using SMART-MDP).

that reduce link overuse, SMART-MDP can satisfy significantly more bandwidth than MDP. This improvement is even more noticeable for higher load levels (the graphical results are not shown due to space limitations). Even under high load (200 Erlangs), SMART-MDP can satisfy approximately 98% of the requested bandwidth.

As we described in Sec. III-D, over-provisioning requires us to allocate additional bandwidth. However, Fig. 7 shows that at higher loads, the option to over-provision yields significant performance improvements. The results verify that the additional resources used with over-provisioning are a good trade-off for the performance gains achieved.

V. Conclusion

We showed that: 1) protecting a fraction of the requested bandwidth, rather than the full amount, is a resource-efficient approach to provisioning connection requests; and 2) the amount of partial-protection that can be guaranteed is limited by both the size of the minimum edge cut between the requested source and destination, and the bandwidth request.

We proposed two online multipath algorithms that guarantee the maximum possible partial-protection. The first algorithm, MDP, uses a simple multipath approach, while the more effective, SMART-MDP algorithm improves on MDP by limiting the overuse of popular links. Our results show that by more effectively utilizing resources, SMART-MDP achieves much lower blocking probability than MDP. Both MDP and SMART-MDP significantly outperform a full-protection approach. Our results show that there is a trade-off between full-protection and performance. Although our algorithms do not provide full-protection, they use fewer network resources, and can therefore satisfy significantly more bandwidth than a full-protection approach. Therefore, for settings in which full-protection is not crucial, our approach would be beneficial.

References

[1] R. Bhandari, Survivable Networks: Algorithms for Diverse Routing. Springer, 1999.

[2] A. Chakrabarti and G. Manimaran, Reliability Constrained Routing in QoS Networks. IEEE/ACM Transactions on Networking, vol. 13, no. 3, pp. 662-675, June 2004.

[3] B. Doshi, S. Dravida, P. Harshavardhana, O. Hauser, and Y. Wang, Optical Network Design and Restoration. Bell Labs Technical Journal, vol. 4, pp. 58-84, Mar. 1999.

[4] J. Fang, M. Sivakumar, A. Somani, and K. Sivalingam, On Partial Protection in Groomed Optical WDM Mesh Networks. In Proceedings of the International Conference on Dependable Systems and Networks, 2005.

[5] A. Fumagalli, I. Cerutti, and M. Tacca, Optimal Design of Survivable Mesh Networks Based on Line Switched WDM Self-Healing Rings. IEEE/ACM Transactions on Networking, vol. 11, no. 3, pp. 501-512, June 2003.

[6] D. Gao and H. Zhang, Routing Pre-Configuration for Fast and Scalable Path Restoration in DWDM Networks. Photonic Network Communications, vol. 12, no. 3, pp. 321-327, Dec. 2006.

[7] W. Grover, Mesh-Based Survivable Networks: Options and Strategies for Optical, MPLS, SONET and ATM Networking. Prentice Hall PTR, Upper Saddle River, New Jersey, 2003.

[8] L. Guo, H. Yu, and L. Li, A New Path Protection Algorithm for Meshed Survivable Wavelength-Division-Multiplexing Networks. Networking ICN 2005, vol. 3420, pp. 68-75.

[9] S. Huang, B. Mukherjee, and C. Martel, Survivable Multipath Provisioning with Differential Delay Constraint in Telecom Mesh Networks. Proceedings of IEEE Infocom. April 2008.

[10] R. Iraschko and W. Grover, A Highly Efficient Path-Restoration Protocol for Management of Optical Network Transport Integrity. IEEE Journal on Selected Areas in Communications, vol. 18, no. 5, pp. 779-794, May 2000.

[11] S. Norden, M. Buddhikot, M. Waldvogel, and S. Suri, Routing Bandwidth-Guaranteed Paths with Restoration in Label-Switched Networks. Proceedings of Computer Networks, vol. 46, no. 2, pp. 197-218, 2004.

[12] C. Ou, L. Sahasabuddle, K. Zhu, C. Martel, and B. Mukherjee, Surviable Virtual Concatenation for Data Over SONET/SDH in Optical Transport Networks. IEEE/ACM Transactions on Networking, vol. 14, pp. 218-231, Feb. 2006.

[13] S. Ramamurthy, L. Sahasrabuddhe, and B. Mukherjee, Survivable WDM Mesh Networks, IEEE/OSA Journal of Lightwave Technology, vol. 21, no. 4, pp. 870-883. April 2003.

[14] R. Roy and B. Mukherjee, Degraded-Service-Aware Multipath Provisioning in Telecom Mesh Networks. Proceedings of IEEE/OSA Optical Fiber Communications Conference, San Diego, CA, Feb. 2008.

[15] H. Wang, E. Modiano, and M. Medard, Partial Path Protection for WDM Networks: End-to-End Recovery Using Local Failure Information. In Proceedings of the Seventh International Symposium on Computers and Communications, 2002.

[16] G. Xue, W. Zhang, J. Tang, and K. Thulasiraman, Establishment of Survivable Connections in WDM Networks Using Partial Path Protection. In Proceedings of IEEE International Conference on Communications, May 2005.

[17] W. Yao and B. Ramamurthy, Survivable Traffic Grooming with Path Protection at the Connection Levelin WDM Mesh Networks. Journal of Lightwave Technology, vol. 23, no. 10, pp. 2846-2853. Oct. 2005.

[18] ITU-T Recommendation, Network Node Interface for the Synchronous Digital Hierarchy (SDH). ITU-T Recommendation G.707, Dec. 2003.